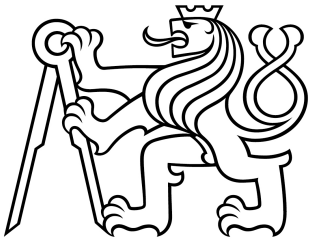
České vysoké učení technické v Praze

Fakulta elektrotechnická

Katedra počítačové grafiky a interakce



Bakalářská práce

Georeferencovaná rozšířená realita

Dominik Truong

Vedoucí práce: Ing. David Sedláček, Ph.D.

Studijní program: Otevřená informatika

Obor: Počítačové hry a grafika

Leden 2020

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Truong** Jméno: **Dominik** Osobní číslo: **465987**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačové grafiky a interakce**

Studijní program: **Otevřená informatika**

Studijní obor: **Počítačové hry a grafika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Georeferencovaná rozšířená realita

Název bakalářské práce anglicky:

Georeferenced Augmented Reality

Pokyny pro vypracování:

Seznamte se s aktuálním stavem projektu EduARd (rozšířená realita pro školy) a s rozšířenou realitou, využívající GPS lokalizaci. Proveďte rešerši aplikací využívajících GPS lokalizaci v rozšířené realitě. Prostudujte možnosti technologie Google - ARCore a projektu ARCore-Location.
Navrhněte a implementujte klientskou aplikaci na mobilním zařízení schopnou zobrazovat data v georeferencované rozšířené realitě v návaznosti na data, která poskytuje server projektu EduARd (formáty X3D, json, rastrové obrázky a video soubory). Implementujte jednosměrnou synchronizaci dat se serverem (stahování dat) a sestavení scény dle staženého scénáře (.json předpis). Implementujte načítání vybraných uzlů formátu X3D (dle dohody s vedoucím práce). Postupujte dle metodiky UCD (User Center Design) pro návrh a testování uživatelského rozhraní s ohledem na dostupnost cílové skupiny.

Seznam doporučené literatury:

[1] Moderní počítačová grafika: Bedřich Beneš, Jiří Sochor, Petr Felkel, Jiří Žára, 2005, Computer press
[2] Augmented Reality: Principles and Practice: Dieter Schmalstieg, Tobias Höllerer, 2016, Addison Wesley
[3] Practical Augmented Reality: Steve Aukstakalnis, 2017, Addison Wesley
[4] T. Lowdermilk, User-Centered Design, O'Reilly Media, 2013
[5] X3D: Web 3D Consortium, [online], http://www.web3d.org/
[6] ARCore location, [online], https://github.com/appoly/ARCore-Location
[7] EduARd: David Sedláček, [online], https://gitlab.fel.cvut.cz/sedlad1/EduARd

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. David Sedláček, Ph.D., katedra počítačové grafiky a interakce FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **13.02.2019** Termín odevzdání bakalářské práce: _____

Platnost zadání bakalářské práce: **20.09.2020**

Ing. David Sedláček, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Ing. Pavel Ripka, CSc. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací.
Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. _____
Datum převzetí zadání

_____
Podpis studenta

Poděkování

Tímto bych chtěl poděkovat Ing. Davidu Sedláčkovi, Ph.D. za vedení mé práce a za cenné informace a rady poskytnuté během práce na tomto projektu. Dále bych chtěl poděkovat své rodině za možnost studovat na vysoké škole. A nakonec bych také chtěl poděkovat své přítelkyni Natálii za neochvějnou podporu a útěchu při studiu.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze 7. ledna 2020

.....

Abstrakt

Tato práce popisuje vývoj mobilní aplikace s rozšířenou realitou využívající GPS lokalizaci. Aplikace načítá 3D scény v rámci scénáře vytvořeného editorem scén. Veškerá potřebná data si aplikace stahuje ze serveru. Aplikace byla vyvíjena v rámci projektu EduARd. Cílovou skupinou jsou především studenti základních a středních škol. Z tohoto důvodu byla aplikace vyvíjena s důrazem na jednoduchost a srozumitelnost. Na základě provedené rešerše byl pro vývoj zvolen herní engine Unity (verze 2019.2.9f1) společně s SDK Vuforia (verze 8.5.8). Aplikace je kompatibilní se zařízeními s operačním systémem Android 4.1 a vyšší.

Klíčová slova: rozšířená realita, AR, GPS lokalizace, GPS, georeferencování, mobilní aplikace, Unity, Vuforia, Android

Abstract

This thesis describes the development of a location-based augmented reality application for mobile devices. The application loads 3D scenes within scenarios created by the scene editor. The application downloads all of the necessary data from the server. This application was developed as a part of project EduARd. The target audience is mainly the students of elementary and high schools. With this in mind, this application was developed with an emphasis on simplicity and comprehensibility. Based on the conducted research the game engine Unity (version 2019.2.9f1) was chosen alongside with the Vuforia SDK (version 8.5.8). The application is compatible with devices using the Android OS 4.1 or newer.

Key words: augmented reality, AR, GPS localization, GPS, location-based, GPS-based, georeferencing, mobile application, Unity, Vuforia, Android

Title translation: Georeferenced Augmented Reality

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

1.1 Motivace

V uplynulém desetiletí zaznamenal svět informačních technologií značně velký pokrok. Znamenitý nárůst výpočetního výkonu pomohl v rozletu mnoha různým odvětvím, od umělé inteligence přes kybernetiku až po počítačovou grafiku. I malá zařízení typu našich telefonů či tabletů jsou dnes vybavena natolik vyspělými komponenty, že vykreslování složitých trojrozměrných scén či natáčení videí ve vysokém rozlišení při zachování 60 snímků za sekundu již nepředstavuje takřka žádný problém. Jednou z technologií, která se v posledním desetiletí významně posunula vpřed je i rozšířená realita. Cílem rozšířené reality je reálný svět tzv. augmentovat, tedy modifikovat a vylepšit. Aplikace s rozšířenou realitou se v posledních letech začínají dostávat do povědomí uživatelů, svá uplatnění nacházejí například v automobilovém, zdravotním a zábavním průmyslu, v armádě či ve školství.

Projekt EduARd, jehož součástí je tato aplikace, si klade za cíl vytvořit systém pro tvorbu virtuálních a naučných stezek pro žáky základních a středních škol. EduARd umožní učitelům tvorbu učebních materiálů se zajímavým obsahem, například úlohami ve formě 3D scén. Tyto scény si žáci budou moci stáhnout do svých zařízení a spustit je v lokalitách definovaných GPS[1] souřadnicemi. Snahou a motivací projektu je ozvláštnit každodenní výuku ve školách skrze interaktivní úlohy, z nichž některé přenesou žáky ven do ulic měst a přírody. Projekt EduARd je vyvíjen týmem profesorů a studentů na Fakultě elektrotechnické Českého vysokého učení technického v Praze.

1.2 Cíle projektu a cílová skupina

Cílem této bakalářské práce je vytvořit návrh a implementaci mobilní klientské 3D aplikace s rozšířenou realitou využívající GPS lokalizaci. Aplikace bude načítat scény v rámci scénáře vytvořeného editorem. Veškeré potřebné soubory popisující danou scénu, včetně 3D objektů nacházejících se ve scéně, bude aplikace stahovat ze serveru. Aplikace bude komunikovat s aplikací pro zobrazení výukových úloh, od níž bude získávat příkazy, které bude vykonávat.

Cílovou skupinou jsou žáci základních a středních škol, tedy osoby ve věku přibližně od 6 do 18 let. Dále je nutné do cílové skupiny zahrnout i učitele těchto škol, jejichž věkovou kategorii nelze s určitostí odhadnout. S ohledem na tento fakt bude aplikace vyvíjena s důrazem na jednoduchost, především pak v uživatelském rozhraní, které by mělo být snadno pochopitelné a srozumitelné.

1.3 Struktura práce

Práce je členěna do kapitol, sekcí a podsekcí. Každá kapitola se bude zabývat určitou fází vývoje projektu. Začátek práce je věnován terminologii, tedy vysvětlením důležitých

[1] Globální polohový systém, angl. Global Positioning System (viz sekce 2.4).

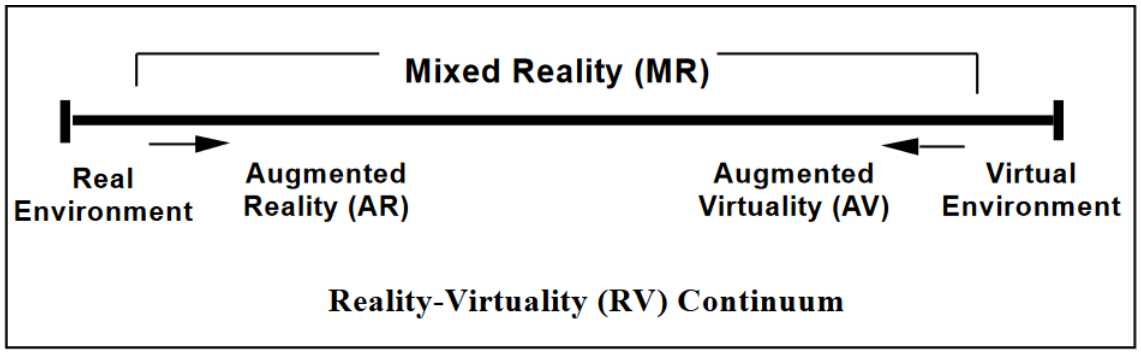
pojmů vyskytujících se v textu. Dále se práce zabývá analýzou poskytující především rešerši dostupných nástrojů a náhled do již existujících projektů zabývajících se podobnou problematikou. Poté je uveden návrh implementace popisující strukturu a funkční princip vyvíjené aplikace společně s návrhem implementace jednotlivých částí. V navazující kapitole o implementaci jsou poté popsány metody realizace projektu v závislosti na předešlém návrhu. Poslední kapitoly se zabývají nejprve testováním aplikace s uživateli a nakonec závěrem shrnujícím průběh vývoje práce, její kvality a nedostatky a budoucí vývoj.

2 Terminologie

Tato kapitola se bude zabývat definicemi termínů klíčových pro tuto práci. Budou zde vysvětleny pojmy jako rozšířená realita, virtuální realita nebo globální polohový systém.

2.1 Realitně-virtuální kontinuum

Pojem realitně-virtuální (RV) kontinuum byl poprvé definován roku 1994 Paulem Milgramem, Haruo Takemurou, Akirou Utsumi a Fumio Kishinem [26]. RV kontinuum představuje znázornění vztahu mezi reálným a virtuálním světem a technologiemi propojujícími oba světy dohromady. Reálný svět je definován jako prostředí s reálnými objekty, které se striktně řídí zákony fyziky. Oproti tomu virtuální svět zákony fyziky respektovat nemusí a je naplněn počítačem vytvořenými virtuálními objekty. Tyto světy tvoří hranice kontinua. Všechny ostatní technologie uvnitř kontinua jsou kategorizovány jako technologie tzv. smíšené reality. Ta je mostem mezi reálným a virtuálním světem, jejími zástupci jsou rozšířená realita a rozšířená virtualita. RV kontinuum je znázorněno na obrázku 2.1.



Obrázek 2.1: Realitně-virtuální kontinuum[1].

2.2 Rozšířená realita

2.2.1 Definice a základní dělení

Rozšířená realita (augmented reality, AR) je souborem technologií umožňujících modifikaci reálného světa přidáním virtuálních prvků generovaných počítačem. Jejich propojením se snaží vylepšit vnímání reálného světa. První zmínky o rozšířené realitě se objevily již v 60. letech minulého století, avšak pojem rozšířená realita byl poprvé použit až v roce 1990 v práci Thomase Caudella a Davida Mizella [19].

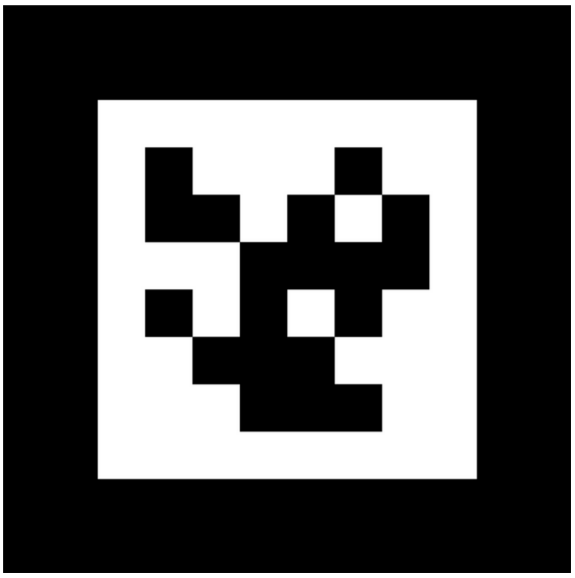
Rozšířenou realitu lze dělit do dvou základních kategorií:

- **Marker-based** - Zařízení detekuje tzv. marker (značku). Značky jsou často černobílé, mohou však být i barevné. Nutnou podmínkou je, aby značky byly dobře detekovatelné v prostředí a jednoznačně rozlišitelné mezi sebou. V případě detekce

[1] Převzato z [26].

jsou virtuální prvky zobrazeny v místě detekované značky. Příklad použitelné značky je zobrazen na obrázku 2.2.

- **Markerless** - Virtuální prvky zobrazuje zařízení na určitých GPS souřadnicích, nebo využívá technologií pro mapování prostředí, například SLAM[2].



Obrázek 2.2: Příklad značky (marker)[3].

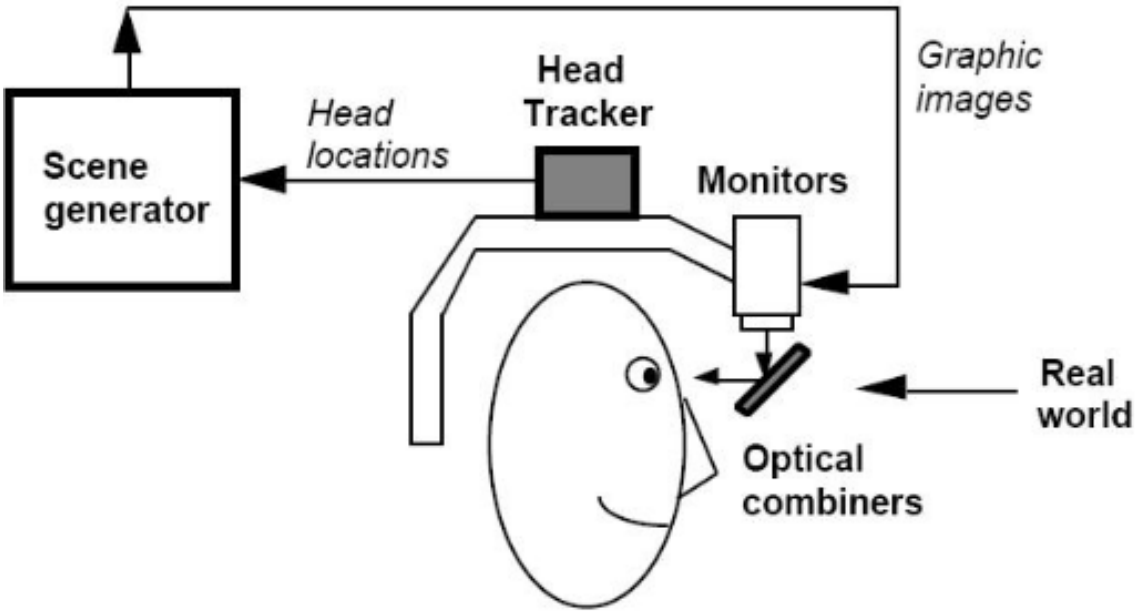
2.2.2 Princip funkce

Zařízení schopná zobrazit rozšířenou realitu se zpravidla skládají z několika základních komponent - kamer a senzorů, výpočetního zařízení a zrcadel či displejů. V některých případech se také uvádí přítomnost projektoru. Úlohou kamery a senzorů je snímat okolní svět a předávat tato data ke zpracování výpočetnímu zařízení. Výpočetní zařízení zpracovává obraz reálného světa získaného ze senzorů a přidává k němu prvky virtuálního světa, například 3D objekty nebo obrázky. Zobrazení virtuálních prvků zajišťují buďto zrcadla nebo displej [13]. Podle technologie zobrazení virtuálních prvků se zařízení dělí do dvou kategorií:

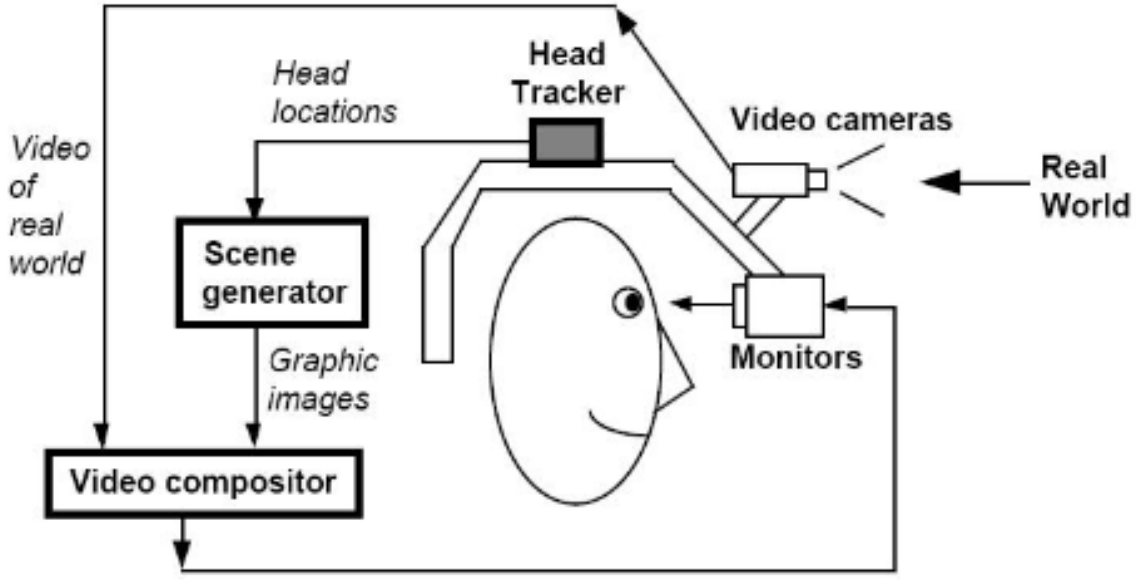
- **Optical See-Through** - Virtuální prvky jsou zobrazeny skrze monokulární či binokulární optický systém umožňující překrývání reálného světa a virtuálních objektů [23].
- **Video See-Through** - Virtuální prvky a obraz reálného světa jsou spojeny do jednoho obrazu, který je uživateli zobrazen skrze displej [23].

Koncepční diagramy obou zařízení jsou znázorněny na obrázcích 2.3 a 2.4.

[2]Simultánní lokalizace a mapování, angl. simultaneous localization and mapping.
[3]Převzato z `https://www.realglitch.com/2013/04/augmented-reality-with-edrawing-on-ipad/`.



Obrázek 2.3: Koncepční diagram zařízení Optical See-Through[4].



Obrázek 2.4: Koncepční diagram zařízení Video See-Through[5].

[4]Převzato z https://pdfs.semanticscholar.org/5b17/a26c79abdc5052ea69ed6ecddd82cbf178bf.pdf.

[5]Převzato z https://pdfs.semanticscholar.org/5b17/a26c79abdc5052ea69ed6ecddd82cbf178bf.pdf.

2.3 Virtuální realita

2.3.1 Definice

Virtuální realita (virtual reality, VR) představuje technologie umožňující tvorbu virtuálních světů, tedy umělých, počítačem generovaných prostředí. Uživatel se zpravidla může v těchto světech pohybovat a interagovat s objekty ve svém okolí. Virtuální světy mohou, ale nemusí respektovat zákony fyziky. Cílem virtuální reality je dostat uživatele do stavu tzv. úplného ponoření či imerze (total immersion). Jedná se o stav, kdy je smyslový prožitek uživatele natolik reálný, že nedokáže rozeznat virtuální svět od reálného [14]. Virtuální realita nachází svá využití v mnoha odvětvích, především pak v těch rizikových, například ve zdravotnictví, letectví či armádě. Simulace nebezpečných situací ve virtuální realitě poskytuje možnost vyzkoušet různé postupy řešení a následně aplikovat ten správný v reálném světě. Příkladem může být složitá operace, nouzové přistání či zneškodnění bomby.

2.3.2 Princip funkce

Ve virtuální realitě se pro zobrazení virtuálního obsahu využívá náhlavních displejů (head-mounted display, HMD). Tato zařízení se skládají především z výpočetních jednotek, množství senzorů a čoček a displeje. Senzory detekují uživatelovy pohyby a aktualizují podle nich virtuální obsah. Pokud tedy uživatel například otočí hlavou v reálném světě, změny se projeví i v tom virtuálním. Mezi základní senzory patří magnetometr, akcelerometr a gyroskop. Magnetometr měří velikost a směr magnetického pole Země a určuje jím tak azimut, tedy úhel, který svírají určitý směr a sever. Gyroskop je elektronické zařízení měřící úhlovou rychlost. Gyroskop měří tzv. Coriolisovu sílu, tedy setrvačnou sílu působící na volně se pohybující objekty v rotující soustavě. Akcelerometr je zařízení schopné měřit posuvné a rotační zrychlení. Pomocí akcelerometru lze vypočítat relativní pozici vůči počátečnímu bodu [25]. Úlohou čoček je měnit obraz jednotlivě pro každé oko v závislosti na tom, kolik obrazu z reálného světa by dané oko vidělo [14]. Displeje pak uživateli zobrazí výsledný virtuální obsah.

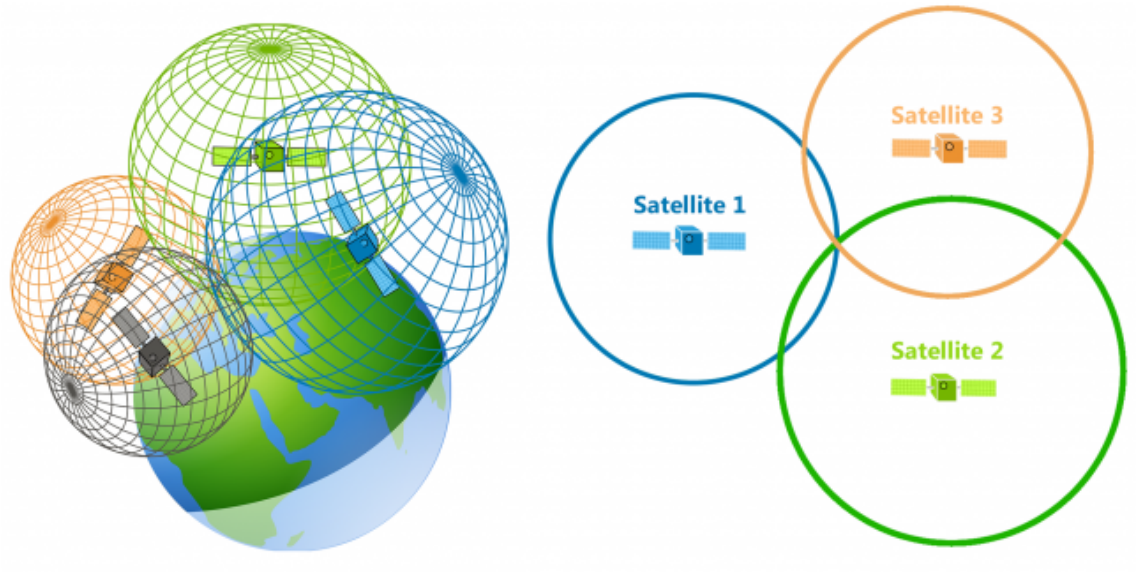
Rozšířená virtualita

Rozšířená virtualita (augmented virtuality, AV) se zabývá technologiemi obohacujícími virtuální světy o prvky reálného světa. V praxi se může například jednat o implementaci ovládání prvků virtuálního světa pomocí rukou. Jedním ze zástupců těchto technologií je například Leap Motion.

2.4 Globální polohový systém

Globální polohový systém (Global Positioning System) neboli GPS je síť 24 satelitů orbitujících okolo Země ve výšce přibližně 20 350 km. Každý ze satelitů posílá údaje o své poloze a času každých 12 hodin, tedy dvakrát denně. Tyto informace jsou posílány skrze rádiové signály pohybujícími se rychlostí světla . Přijímače (GPS lokátory, mobilní zařízení, navigace apod.) dokáží na základě těchto signálů vypočítat svou vzdálenost od daného satelitu [7].

K výpočtu polohy přijímače se využívá procesu zvaného trilaterace (viz obrázek 2.5). Oproti triangulaci, která využívá úhlů k určení polohy, trilaterace využívá vzdáleností k určení polohy. Pokud přijímač zná svou vzdálenost od jednoho satelitu, dokáže určit sféru, na níž leží . K určení polohy je zapotřebí nejméně tří satelitů (čtyř pokud určujeme i nadmořskou výšku), výsledná poloha je získána výpočtem průniku tří sfér [6]. Obecně platí, že čím více signálů přijímač zachytí, tím přesnější je výpočet polohy. V praxi se však ukazuje, že přesnost GPS může být ovlivněna mnohými dalšími faktory, například atmosférickými podmínkami, přesností hodin v přijímači, polohou přijímače, elektromagnetickými poli nebo odrazem samotného signálu [27].



Obrázek 2.5: Trilaterace[6].

[6]Převzato z `https://gisgeography.com/trilateration-triangulation-gps/`.

3 Analýza

Tato kapitola se bude zabývat analýzou projektu. Nejprve se bude zabývat rešerší dostupných SDK[1] pro rozšířenou realitu a platforem pro tvorbu aplikací využívajících GPS lokalizaci v rozšířené realitě. Dále zde budou představeny další aplikace podobné aplikaci popisované v této práci. Nakonec bude poskytnuto shrnutí vysvětlující výběr konkrétního SDK použitého pro vývoj této aplikace.

3.1 Dostupná SDK

3.1.1 Vuforia

Vuforia je SDK pro tvorbu aplikací s rozšířenou nebo smíšenou realitou. Vuforia je multiplatformní, podporující vývoj aplikací pro Android (4.1 a vyšší), iOS (9.0 a vyšší) či Universal Windows Platform. Vuforia nabízí API[2] v programovacích jazycích Java, C++, Objective C++, a .NET [1]. Od verze 2017.2 je Vuforia integrována do herního enginu Unity.

Mezi její základní vlastnosti patří detekce a sledování 2D a 3D objektů. V případě 2D objektů se jedná o detekci značek (marker, viz sekce 2.2.1) a tzv. VuMarks, což jsou speciální čárové kódy v podobě obrázků. Z 3D objektů dokáže Vuforia detekovat některá základní primitiva, například kvádr, válec či kužel. Vuforia dále podporuje detekci více objektů zároveň (multi target tracking), vykreslování virtuálních prvků i při ztrátě značky (extended tracking) či vykreslování objektů bez použití značek (markerless tracking) [1]. Vuforia také nabízí možnost ukládání objektů na cloudové uložiště. Vuforia nabízí čtyři typy licencí: Free, Basic, Basic+Cloud a Pro. Licence se mezi sebou liší především počtem funkcí a cenou.

3.1.2 ARCore

ARCore je SDK umožňující tvorbu aplikací s rozšířenou realitou. ARCore je vyvíjen společností Google, je dostupný na vybraných zařízeních s operačními systémy Android (7.0 a vyšší) a iOS (11.0 a vyšší)[3]. Také je integrován v herním enginu Unity. Pro zobrazení virtuálního obsahu v reálném světě využívá ARCore tří základních technologií: sledování pohybu zařízení (motion tracking), pomocí něhož dokáže určit polohu zařízení vzhledem k prostředí (world tracking), porozumění prostředí (environmental understanding), které slouží k detekci polohy a velikosti vertikálních, horizontálních a šikmých ploch, a odhad světla (light estimation) [1]. ARCore podporuje jak detekci a sledování 2D značek, tak zobrazování virtuálních prvků bez značek. K tomu využívá technologie SLAM, pomocí níž mapuje okolí zařízení, do něhož vykresluje virtuální obsah. ARCore je poskytován zcela zdarma.

[1] Soubor nástrojů pro vývoj softwaru, angl. software development kit.

[2] Rozhraní pro programování aplikací, angl. application programming interface.

[3] Seznam podporovaných zařízení viz `https://developers.google.com/ar/discover/supported-devices`.

3.1.3 ARKit

ARKit je SDK pro tvorbu rozšířeně realitních aplikací pro vybraná zařízení s operačním systémem iOS. Jedná se o zařízení iPhone 6s a vyšší využívající procesory A9, A10 nebo A11[4]. Základními vlastnostmi tohoto SDK jsou především porozumění prostředí a schopnost detekovat horizontální a vertikální plochy a kontextuálně správně na ně umisťovat virtuální objekty (tedy například pokládat objekty na stůl místo podlahy) [1]. ARKit využívá technologie SLAM, dokáže odhadnout intenzitu světla v prostředí a od verze 2.0 podporuje i detekci a sledování 2D značek. Verze 3.0 dále přináší funkcionality typu zaznamenávání pohybu (motion capture), detekce a sledování několika obličejů zároveň (multiple face tracking), okluze lidí, využití přední a zadní kamery zároveň pro detekci a sledování objektů ve světě [5]. ARKit je integrován do herního enginu Unity a je poskytován zdarma.

3.1.4 Wikitude

Wikitude je SDK původně vycházející z aplikace Wikitude World Browser. Původním cílem tohoto projektu bylo dodat na trh platformu umožňující vývoj aplikací s rozšířenou realitou využívajících GPS lokalizaci. Tato funkcionalita byla zachována i v dalších verzích SDK a je dostupná i v nynější verzi. Dalšími funkcemi tohoto SDK jsou detekce a sledování 2D značek a 3D objektů, sledování více značek najednou, vykreslování virtuálního obsahu bez použití značek, vykreslení virtuálního obsahu i při ztrátě značky, ukládání značek na cloudové uložiště, detekce ploch a mapování prostředí pomocí technologie SLAM apod. Wikitude navíc využívá technologie SMART[5], která umožňuje přepínání mezi SDK ARCore a ARKit na zařízeních, která je podporují [12]. Díky tomu je Wikitude dostupné na větším počtu zařízení jak s operačními systémy Windows, Android (4.4 a vyšší) či iOS (9.0 a vyšší), tak i na tzv. chytrých brýlích. Wikitude je též integrováno do herního enginu Unity, dále je podporováno frameworky[6] Xamarin, Cordova či Flutter [22]. Wikitude je placené SDK, nabízí jednorázově placené licence začínající na 1990 eurech a roční licence začínající na 2490 eurech. Oproti jednorázovým licencím nabízejí roční licence především aktualizace SDK, větší množství dostupných služeb a lepší podporu ze strany tvůrců a vývojářů. Je možné si zažádat o licenci pro malé firmy a nezávislé vývojáře, která je poskytována zdarma. Pro kladné vyřízení však žadatel musí splnit seznam daných požadavků.

3.1.5 AR Foundation

AR Foundation je balíčkem (package) pro herní engine Unity. Cílem tohoto projektu je vytvořit platformu pro tvorbu aplikací s rozšířenou realitou spojením dvou SDK dohromady: ARCore a ARKit. Tímto spojením nabízí podporu pro větší množství zařízení jak pro Android, tak i iOS. Snahou AR Foundation je implementace funkcí společných pro obě zmíněná SDK. V nynější verzi AR Foundation nabízí detekci a sledování 2D a 3D objektů, detekci obličejů, určení polohy a orientace zařízení v prostoru, detekci horizontálních a vertikálních ploch, detekci významných bodů (feature points) či schopnost odhadu světla [17]. Tento balíček je nabízen zdarma, lze ho však použít pouze prostřednictvím Unity.

[4]Seznam podporovaných zařízení viz `https://www.redmondpie.com/ios-11-arkit-compatibility-check-if-your-device-is-compatible-with-apples-new-ar-platform/`.

[5]Angl. seamless augmented reality tracking.

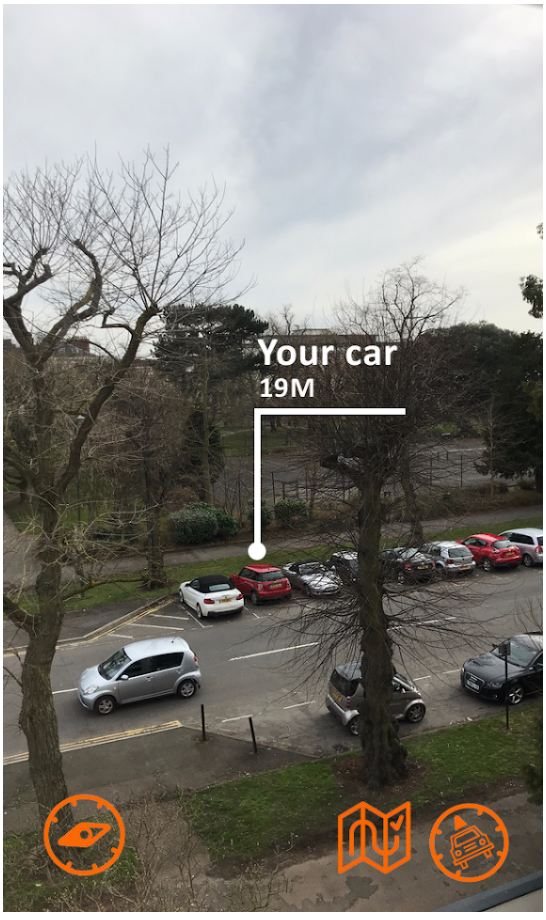
[6]Aplikační rámec sloužící k vývoji jiných aplikací či softwarových projektů.

3.2 Platformy pro tvorbu aplikací s GPS lokalizací

3.2.1 ARCore Location

ARCore Location je sada nástrojů (toolkit) pro tvorbu rozšířeně realitních aplikací využívajících GPS lokalizaci. Jedná se o projekt vyvíjený společností Appoly. Projekt vychází z ARCore SDK (viz sekce 3.1.2), je tedy dostupný pro vybraná zařízení s operačními systémy Android a iOS. Cílem projektu je vytvořit nástroje pro vykreslování objektů v lokacích daných GPS souřadnicemi. ARCore Location využívá tzv. anchors (kotev), které slouží k ukotvení objektů v prostoru. Kotvy zaručují, že objekty jim přidružené si zachovají svou pozici a rotaci i v případě, že se se zařízením bude hýbat, například při chůzi či otáčení [2]. Příkladem aplikace, která vznikla využitím této platformy je Where's my cAR? (viz obrázek 3.1).

Jelikož je tento projekt stále v raném stádiu, vývojáři přiznávají, že se potýkají s několika signifikantními problémy, především hardwarového charakteru. Jedním z problémů je nepřesnost kompasu, která je v rozmezí 15 úhlových stupňů. Tento problém přetrvává i v poslední verzi. Dalším problémem je fixně daná doba aktualizace kotev, nastavená na 8 sekund. Pokud se uživatel pohybuje příliš velkou rychlostí, může se přiblížit objektům, které by od něj měly být dále [3].

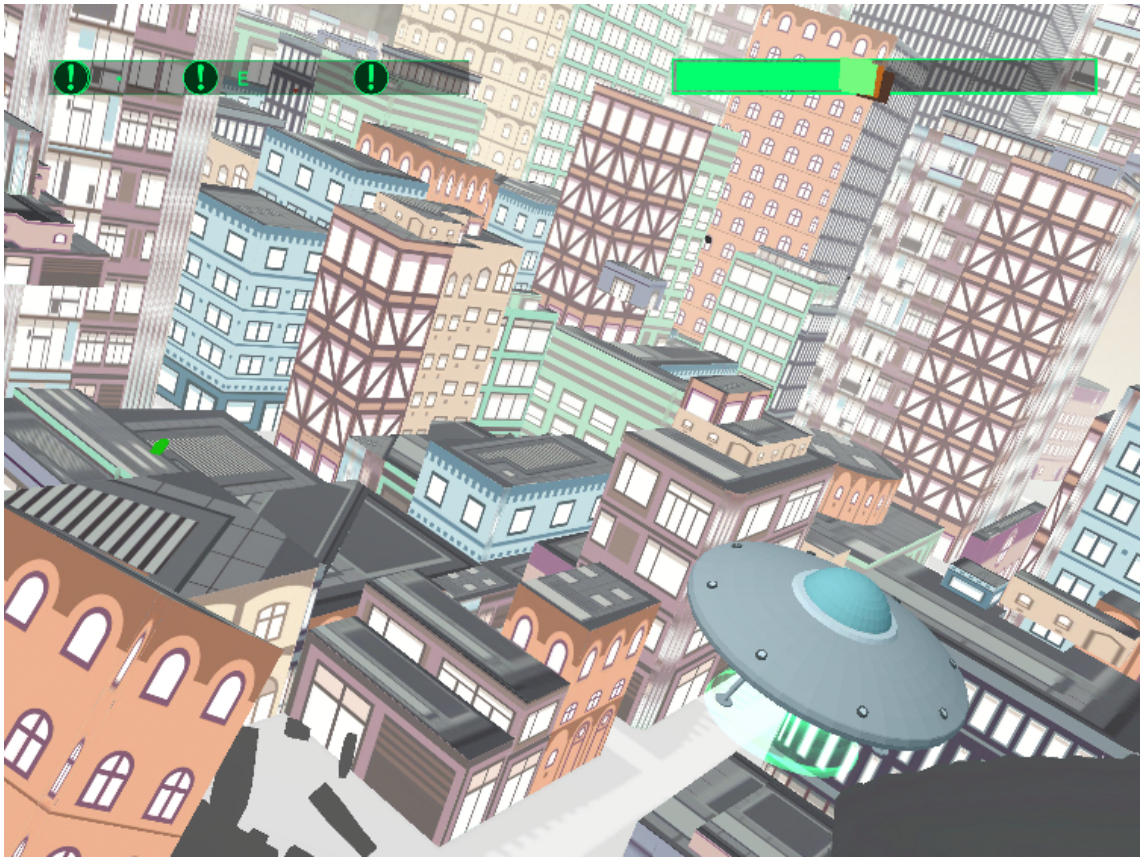


Obrázek 3.1: Aplikace Where's my cAR?[7].

[7]Převzato z `https://play.google.com/store/apps/details?id=uk.co.appoly.wheres_my_car`.

3.2.2 Mapbox

Mapbox je SDK nabízející velké množství online map, včetně navigací a API k webovým službám. Mapbox je multiplatformní, je dostupný například na platformách Android, iOS či herním enginu Unity. Mapbox umožňuje tvorbu aplikací využívajících GPS lokalizaci nejen v rozšířené realitě [10]. Případem použití mohou být turistické aplikace, ukazující zajímavá místa v okolí uživatelovy pozice, navigační aplikace, či RPG[8] a strategické hry. Jednou z aplikací v rozšířené realitě je Space Invad-AR, v níž má hráč možnost ovládat mimozemskou loď přímo v ulicích města (viz obrázek 3.2).



Obrázek 3.2: Hra Space Invad-AR[9].

3.2.3 Unity AR+GPS Location

AR+GPS Location je balíček nástrojů pro Unity. Umožňuje tvorbu aplikací vykreslujících 3D objekty v reálném světě. K tomu využívá GPS lokalizace, tedy zeměpisnou šířku, délku a nadmořskou výšku. AR+GPS Location je postaven na kombinaci SDK Vuforia a ARFoundation (viz kapitoly 3.1.1 a 3.1.5). Díky tomu podporuje zařízení s operačními systémy Android i iOS. Podmínkou je však také podpora zařízení pro ARCore a ARKit [16].

Hlavními funkcionalitami tohoto balíčku jsou, kromě vykreslování objektů dle GPS pozice, tzv. AR Hotspots, tedy zajímavá místa, na která je uživatel upozorněn, nachází-li se v

[8] Hra na hrdiny, angl. role-playing game.

[9] Převzato z `https://blog.mapbox.com/mapbox-unity-sdk-augmented-reality-space-invadar-aa2b329d762e`.

jejich blízkosti. Dále je to vykreslování 3D textu, pohyb objektů podél křivek Catmull-Rom nebo stínování objektů na podlaze. Naopak hlavními limitacemi jsou nepřesnost vykreslování, pokud je zadána nadmořská výška, a nepřesnost GPS lokalizace [16]. Ukázka aplikace vytvořené pomocí AR+GPS Location je na obrázku 3.3.

Obrázek 3.3: Aplikace vytvořená pomocí AR+GPS Location[10].

3.3 Aplikace

Pokémon GO

Pokémon GO je hra z roku 2016 vydaná společností Niantic. Cílem hry je sbírat tvory ze světa Pokémon a využívat je v soubojích mezi ostatními hráči. Hra je dostupná pro platformy Android a iOS, dle statistik má již přes 1 miliardu stažení [11]. Pokémon GO využívá GPS lokalizace, hráči ukazuje Pokémony v jeho blízkém okolí. Pokud se hráč dostaví na jedno z těchto míst, využitím rozšířené reality mu hra zobrazí daného Pokémona v reálném světě. Ukázka ze hry je na obrázku 3.4.

Ingress Prime

Ingress Prime je jednou ze starších rozšířeně realitních her využívajících GPS lokalizaci, která se dostala do povědomí širší veřejnosti. Původně byla vydána roku 2012 pod názvem Ingress společností Niantic, která stojí i za hrou Pokémon GO. Ingress Prime je dostupná na platformách Android a iOS. Cílem hry je zisk vlivu pro hráčem zvolenou frakci. Ten hráč získá útokem na tzv. portály, které se hráči zobrazují na mapě. Aby ne ně mohl zaútočit, musí se dostavit na daná místa v reálném světě. Kromě vlivu získává hráč i zkušenostní body a tzv. exotickou hmotu [8]. Ukázka ze hry Ingress Prime je na obrázku 3.5.

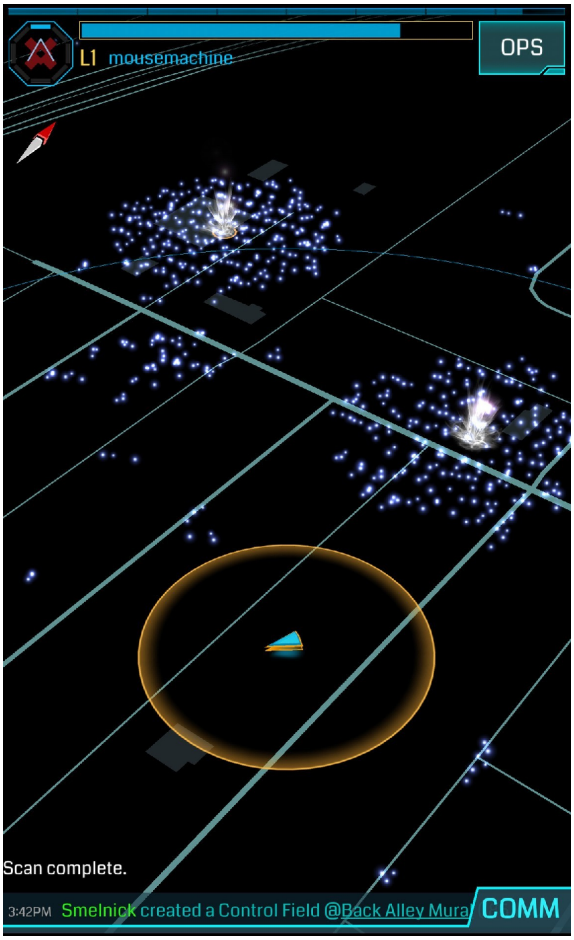
[10]Převzato z `https://assetstore.unity.com/packages/tools/integration/ar-gps-location-134882`.

Obrázek 3.4: Hra Pokémon GO[11].

Vortex Planetarium

Vortex Planetarium je aplikace zobrazující kosmická tělesa pomocí rozšířené reality a GPS lokalizace. Jejím prostředním lze pozorovat přes 20 tisíc různých těles, od souhvězdí, hvězd a planet, až po objekty z Caldwellova či Messierova katalogu [18]. Objekty jsou zobrazovány v závislosti na uživatelově poloze. Aplikace dále umožňuje sledovat oblohu v různých denních dobách, ukazuje zajímavá fakta a nadcházející události. Aplikace je dostupná pouze na platformě Android, její ukázka je na obrázku 3.6.

[11] Převzato z `https://www.androidauthority.com/pokemon-go-arcore-update-913547/`.



Obrázek 3.5: Hra Ingress Prime[12].

Obrázek 3.6: Aplikace Vortex Planetarium[13].

[12]Převzato z [8].
[13]Převzato z `http://www.vortexplanetarium.com/`.

3.4 Shrnutí

Vzhledem k provedené analýze, rešerši a předešlé práci na tomto projektu v rámci semestrálního projektu bylo pro vývoj aplikace zvoleno SDK Vuforia. Projekt EduARd cílí na žáky základních a středních škol, u kterých se nepředpokládá vlastnictví nejnovějších a technicky nejvyspělejších zařízení. Důležitější je kompatibilita s co největším množstvím zařízení, především pak se staršími zařízeními na starších platformách. Oproti svým konkurentům je Vuforia dostupná na poměrně velkém množství zařízení, včetně těch starších (minimálním požadavkem je operační systém Android 4.1 Jelly Bean). Dále je Vuforia v základní verzi dostupná zdarma, pro získání licence je nutné se pouze zaregistrovat na jejích oficiálních stránkách. Výběr tohoto SDK je také umocněn faktem, že v další fázi vývoje této aplikace, která již není zahrnuta v této bakalářské práci, bude nutno dodat detekci 2D značek. Tuto funkcionalitu Vuforia nativně podporuje a již dlouho vyvíjí a vylepšuje. V neposlední řadě je nutné zmínit dobrou komunikaci a podporu ze strany tvůrců a vývojářů, kteří s tímto SDK pracují. Jako dobrou alternativu při vývoji rozšířeně realitních aplikací s využitím GPS lokalizace bych doporučil Wikitude. Wikitude již od samého počátku poskytuje podporu pro vývoj právě těchto aplikací a v provedené rešerši se jedná o jediné SDK, které nativně tuto funkcionalitu podporuje.

4 Návrh

Následující kapitola se bude zabývat návrhem aplikace. Nejprve zde bude uvedena základní charakteristika aplikace a následně bude vysvětlen princip její funkce a popsán návrh jednotlivých částí.

4.1 Základní charakteristika

Jak již bylo uvedeno v úvodu (sekce 1.2) cílem projektu je navrhnout a implementovat klientskou 3D aplikaci s rozšířenou realitou a GPS lokalizací. Cílovou skupinou jsou žáci základních a středních škol a učitelé působící na těchto školách. Základní vlastnosti aplikace a požadavky pro splnění zadání práce jsou uvedeny v následujících bodech.

- **P1** - Aplikace bude načítat scény v rámci scénářů vytvořených editorem 3D scén[1].
- **P2** - Scény se budou zobrazovat pomocí technologií rozšířené reality v reálném světě skrze mobilní zařízení či tablety.
- **P3** - Poloha scény bude určena pomocí GPS souřadnic. Scéna se bude vykreslovat v závislosti na poloze uživatele.
- **P4** - V aplikaci bude implementována jednosměrná komunikace se serverem projektu EduARd (stahování). Ze serveru si bude aplikace stahovat veškerá potřebná data pro načtení scény. Jmenovitě se bude jednat o soubor formátu JSON[2] popisující scény ve scénáři, soubory formátu X3D[3] obsahující definice 3D modelů, rastrové obrázky a videa.
- **P5** - Data budou ukládána lokálně na zařízení pro případné opětovné načtení stejné scény.
- **P6** - Přestože pro stahování dat ze serveru bude nutné internetové připojení, pro následné načtení a vykreslení scény bude nutná pouze GPS poloha zařízení. Aplikace tedy bude schopna funkce i bez připojení k internetu za podmínky, že veškerá nutná data budou již předem stažena do zařízení.
- **P7** - Aplikace bude komunikovat s aplikací pro zobrazení výukových úloh (v tomto textu dále jen jako řídící aplikace). Od ní bude dostávat příkazy, které bude vykonávat, včetně příkazu načtení scény. Aplikace tedy nebude samostatně spustitelná bez příkazů od řídící aplikace.
- **P8** - Uživatel aplikace bude moct v rámci scény interagovat s objekty, které se v ní nachází. V aplikaci bude implementována množina událostí, tedy souborů tzv. spouštěčů a akcí. Pokud uživatel splní podmínky všech spouštěčů (např. stiskne tlačítko na displeji), vyvolají se příslušné akce (např. se spustí animace objektu).

[1] Editor je vyvíjen současně s touto aplikací, avšak ne autorem této práce a není tak její součástí.
[2] Angl. JavaScript Object Notation (viz sekce 5.1.3).
[3] Angl. Extensible 3D (viz sekce 5.1.2).

- **P9** - Aplikace bude kompatibilní se zařízeními s operačním systémem Android. Míra kompatibility s různými verzemi tohoto operačního systému, především pak těmi staršími, bude záviset na výběru SDK. Vzhledem k provedené analýze (viz kapitola 3) bude využito SDK Vuforia.

4.2 Funkce aplikace

Funkci aplikace lze rozdělit do tří fází: fáze stahování/aktualizace dat, fáze načítání a vykreslování scény a fáze mazání dat. Spouštění jednotlivých fází je závislé na příkazech přijatých od řídící aplikace.

- Ve fázi stahování/aktualizace dat stáhne aplikace veškerá potřebná data pro vykreslení scén daného scénáře. Pokud se jedná o aktualizaci, stáhne pouze data, která na zařízení chybí nebo byla aktualizována, přebytečné soubory smaže. Jedná se o jedinou fázi, při které je nutné mít k dispozici internetové připojení. Pokud v této fázi dojde k chybě, například při stahování dat, aplikace o této skutečnosti informuje řídící aplikaci. Na konci této fáze se aplikace vypne a předá kontrolu řídící aplikaci.
- Ve fázi načítání a vykreslování scény načte aplikace nutná data vztahující se k dané scéně. Předpokladem je, že tato data již byla do zařízení stažena. V opačném případě je informována řídící aplikace, která pošle příkaz na stažení dat. Po úspěšném načtení scény aplikace vykreslí objekty podle polohy zařízení. Dále provádí správu událostí, kontroluje spouštěče a spouští akce.
- Ve fázi mazání dat aplikace smaže veškerá data o scénáři. Na konci této fáze se aplikace ukončí a předá kontrolu řídící aplikaci.

4.3 Scénáře a scény

Scénář je souborem scén definujících množinu objektů, jejich vlastnosti a způsoby interakce s nimi. Scénář je výstupem editoru 3D scén, což je webová aplikace umožňující tvorbu a úpravu scénářů. Pro popis scénáře byl vybrán formát JSON. Scénář je hierarchicky uspořádaný do uzlů, každý objekt je jedním reprezentován. Objekty jsou jednoznačně identifikovatelné svým názvem. Typy objektů a popis jejich funkce jsou uvedeny níže, hierarchie objektů je zobrazena na obrázku 4.1.

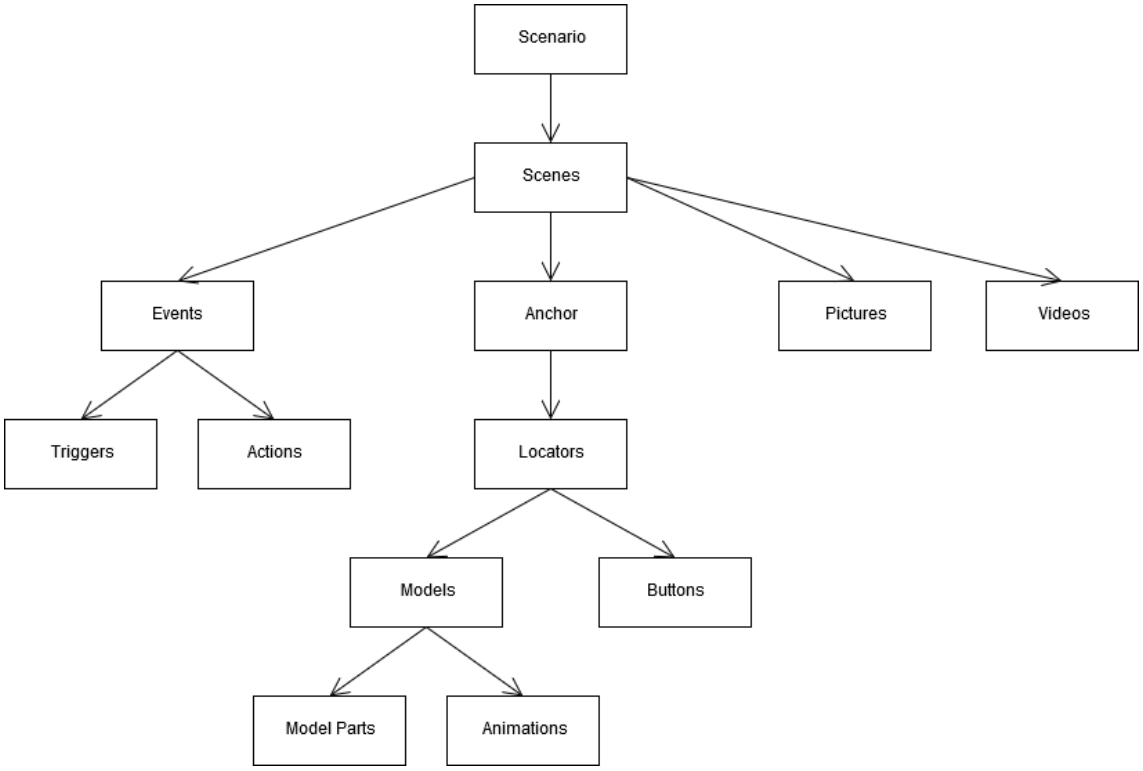
- **Scenario** - Objekt popisující scénář. Obsahuje seznam scén, které může aplikace načíst.
- **Scene** - Objekt popisující scénu. Scéna je seznamem dalších objektů a událostí definujících způsoby interakce s objekty.
- **Anchor** - Objekt popisující kotvu. Kotva reprezentuje počátek scény, je definovaná GPS souřadnicemi, tedy zeměpisnou šířkou, délkou a nadmořskou výškou. Scéna je vykreslována vůči GPS poloze uživatele. V každé scéně se nachází právě jedna kotva.
- **Locator** - Objekt popisující lokátor. Lokátor je pomocný objekt sdružující určitou podmnožinu objektů ve scéně. Lokátory usnadňují manipulaci s více objekty najednou.

- **Model** - Objekt popisující model. Model je 3D objekt ve scéně. Definice modelu je uložena v souboru X3D, nikoliv ve scénáři, kde je uchován pouze odkaz na něj. Ve scénáři je dále uložena poloha modelu vůči počátku scény a jeho velikost. Modely se blíže zabývá sekce 4.4.
- **Model Part** - Objekt popisující část modelu. Modely je vhodné členit do menších částí, pokud je potřeba definovat interakci jen pro tyto dané části.
- **Animation** - Objekt popisující animaci. Animace není přímo definována ve scénáři, je zde pouze uchován odkaz do souboru X3D, který obsahuje definice jak modelů, tak i animací. Typy animací a jejich popisy jsou uvedeny v sekci 4.5.
- **Button** - Objekt popisující tlačítko. Tlačítko je elementem uživatelského rozhraní, jedná se o jeden ze spouštěčů akcí.
- **Event** - Objekt popisující událost. Události definují interakce s objekty ve scéně pomocí seznamu spouštěčů a akcí.
- **Trigger** - Objekt popisující spouštěč. Spouštěče definují podmínky pro spuštění akcí, spouštěčem může být například stisknutí tlačítka.
- **Action** - Objekt popisující akci. Akce jsou spouštěny, pokud jsou podmínky všech spouštěčů splněny.
- **Picture** - Objekt popisující obrázek. Objekt obsahuje pouze odkaz na obrázek, zobrazování obrázků je jednou z akcí.
- **Video** - Objekt popisující video. Objekt obsahuje pouze odkaz na video, přehrávání videí je jednou z akcí.

4.4 Modely

Jelikož je editor 3D scén webovou aplikací, využívá modelů ve formátu X3D, který lze jednoduše zobrazit skrze webové prohlížeče. V souladu s touto volbou je nutné, aby i klientská aplikace dokázala načítat soubory tohoto formátu. Soubory X3D obsahují definice modelů ve scéně, jmenovitě například definice vrcholů, trojúhelníků, materiálů, texturovacích souřadnic (včetně odkazů na samotné textury) či transformací těchto objektů (translace, rotace, škálování). Dále umožňuje definovat animace pomocí interpolátorů, senzorů a cest. Jelikož klientská aplikace načítá pouze modely definované v souborech X3D a nikoliv celou 3D scénu, bude některé uzly při načítání ignorovat (například kameru nebo osvětlení). Příklad jednoduchého modelu definovaného pomocí formátu X3D je uveden níže.

```
<Transform DEF='CUBE'>
    <Shape>
        <Appearance>
            <Material DEF='COLOR' diffuseColor='1 0 0'/>
        </Appearance>
        <Box/>
    </Shape>
</Transform>
```



Obrázek 4.1: Hierarchie objektů ve scénáři

4.5 Interakce

Interakce je ve scénáři definována množinou událostí. Ty definují jednotlivé spouštěče a přiřazují jim jednotlivé akce. Aplikace pravidelně kontroluje, zdali je podmínka všech spouštěčů splněna. Pokud ano, vyvolá přiřazené akce.

Spouštěče

V následující bodech budou uvedeny všechny spouštěče a podmínky jejich splnění. Pro spuštění akce může být zapotřebí splnění podmínek více spouštěčů nebo naopak jeden spouštěč může vyvolat více akcí.

- **Vzdálenost od objektu** - Podmínka je splněna, pokud je uživatel od objektu ve vzdálenosti menší nebo rovné vzdálenosti definované spouštěčem.
- **Pohled na objekt** - Podmínka je splněna, pokud se uživatel dívá na objekt daný spouštěčem.
- **Stisknutí tlačítka** - Podmínka je splněna, pokud uživatel stiskne tlačítko dané spouštěčem.

Akce

V následujících bodech budou uvedeny všechny akce a popis jejich chování.

- **Interpolace barvy** - Změna barvy materiálu v čase. Možnost interpolace difúzní, spekulární a emisivní složky materiálu.

- **Interpolace rotace** - Změna rotace objektu v čase.
- **Interpolace pozice** - Změna pozice objektu (translace) v čase.
- **Interpolace škály** - Změna velikosti (škály) v čase.
- **Interpolace skaláru** - Změna hodnoty skaláru v čase. Možnost interpolace průhlednosti a drsnosti materiálu, poloměru sféry nebo výšky válce.
- **Zobrazování obrázků** - Zobrazení obrázku v rámci uživatelského rozhraní.
- **Přehrávání videí** - Přehrání videa v rámci uživatelského rozhraní.
- **Zviditelnění a zneviditelnění objektů** - Zobrazení objektu nebo jeho zneviditelnění.

5 Implementace

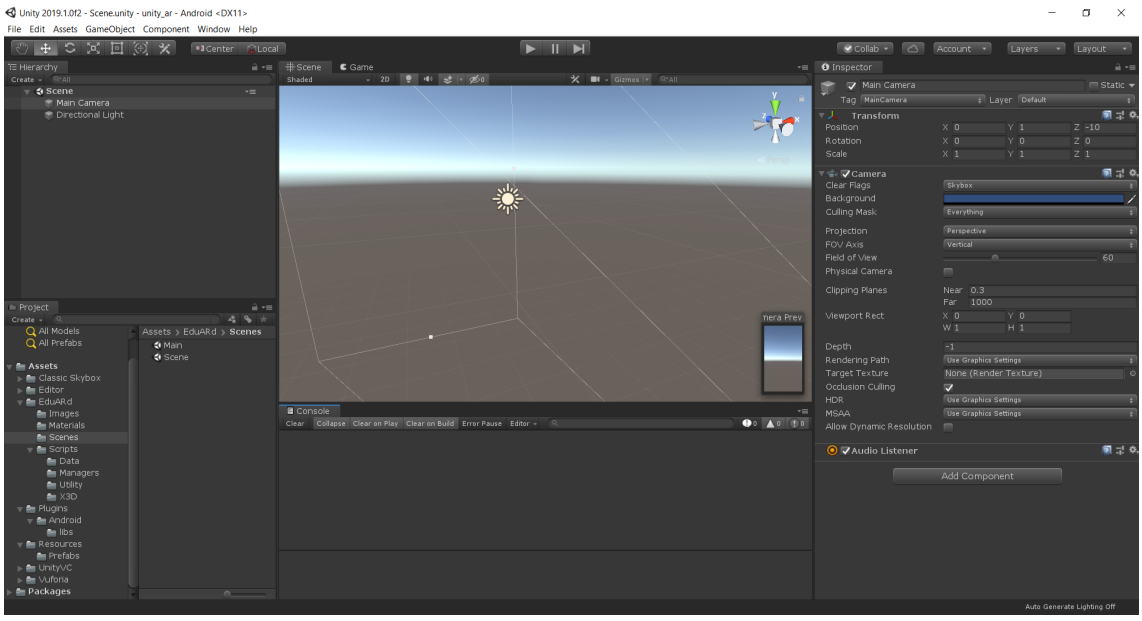
Tato kapitola se bude zabývat implementací projektu. Budou zde popsány metody realizace jednotlivých částí aplikace počínaje uvedením použitých nástrojů. Dále zde budou vysvětleny například způsoby implementace komunikace se serverem a řídící aplikací, GPS lokalizace, načítání scén či implementace uživatelského rozhraní.

5.1 Použité nástroje a technologie

5.1.1 Unity

Unity je multiplatformní herní engine vytvořený společností Unity Technologies. Unity umožňuje vytváření 2D a 3D aplikací s podporou více než 25 platforem, včetně například Microsoft Windows, Linux, Android nebo iOS. Programovacím jazykem Unity je C#, ačkoliv v předešlých verzích byly podporovány i jazyky JavaScript (ve spojitosti s Unity též nazývaný jako UnityScript) a Boo. Vývojářum nabízí Unity velké množství nástrojů a funkcionalit, jmenovitě například podporu normálových map, dynamických stínů či fyzikálních simulací [24].

Unity je vydáváno ve třech verzích: Personal, Plus a Pro. V závislosti na poskytovaných nástrojích a funkcionalitách, obratu jedince či společnosti a podpoře ze strany tvůrců se cena jednotlivých verzí liší. Základní verze Unity Personal je poskytována zdarma. Tento projekt byl vyvíjen ve verzích Unity Plus 2018.2.12f1 až 2019.2.9f1. Ukázka Unity Editoru je na obrázku 5.1.



Obrázek 5.1: Unity Editor

5.1.2 X3D

X3D neboli Extensible 3D je grafický formát vycházející z formátu VRML[1]. Jedná se o třetí verzi formátu VRML, proto je někdy označován jako VRML 3.0. Formát X3D umožňuje reprezentaci 3D scén, mezi jeho základní vlastnosti patří rozšiřitelnost (extensibility), reprezentace scény pomocí stromové struktury, hraniční reprezentace modelů, podpora interakcí (systému senzorů a událostí), podpora animací atd. [21]. Od verze VRML 2 (také označovanou jako VRML 97) je tento formát ISO[2] standardem.

Název X3D vychází ze spojení dvou zkratek - XML[3] a 3D. Pro reprezentaci scény lze tedy použít syntaxi jazyka XML. Výhodou tohoto přístupu je například velké množství knihoven podporujících formát XML či poměrně jednoduchý převod do jiných formátů. Kromě formátu XML lze ještě použít starší, avšak méně častý formát založený na VRML 97 [29].

5.1.3 JSON

JSON neboli JavaScript Object Notation je jednotný formát pro výměnu dat. Mezi jeho hlavní výhody patří lidská i strojová čitelnost. JSON je nezávislý na programovacích jazycích, využívá však konvencí společných s rodinou programovacích jazyků C (např. C/C++, C#, Java nebo JavaScript). JSON je postaven na dvou základních strukturách - seznamu dvojic jméno-hodnota a uspořádaném seznamu hodnot [9]. Původně vznikl jako konkurence formátu XML, své využití našel například jako formát dat, které si mezi sebou vyměňují webové aplikace.

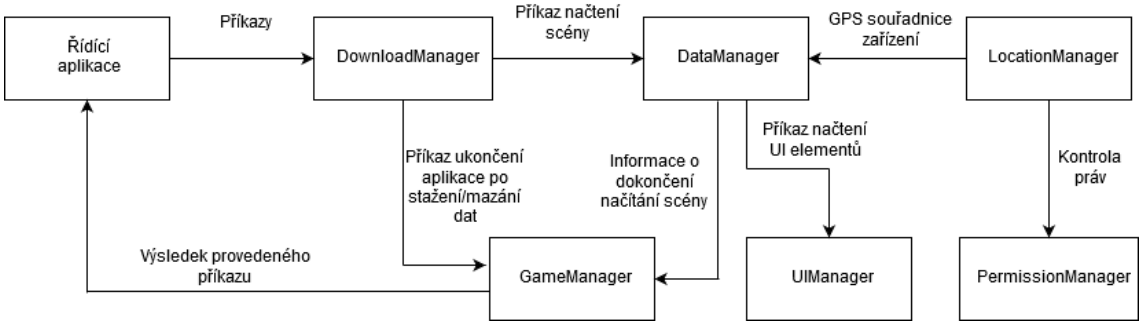
5.2 Manažeři

O správu aplikace se starají tzv. manažeři. Každý manažer je třídou starající se o určitou část aplikace. Manažeři mezi sebou komunikují skrze funkce a události, předávají si tak důležitá data a informace. Manažeři jsou navrženi s cílem co největší nezávislosti na ostatních manažerech. Cílem tohoto způsobu implementace je vytvoření robustního řešení, které je rozumně členěné, srozumitelné a udržovatelné. V aplikaci je implementováno šest manažerů: `DownloadManager`, `DataManager`, `GameManager`, `LocationManager`, `PermissionManager` a `UIManager`. V následujících bodech bude stručně popsána jejich funkce, podrobněji se jimi budou zabývat další sekce v této kapitole. Na obrázku 5.2 je zobrazeno schéma komunikace mezi manažery.

- `DownloadManager` - Zpracovává příkazy od řídící aplikace a spravuje stahování a mazání dat scénářů.
- `DataManager` - Spravuje načítání dat a načítání scény. Po načtení scény aktualizuje její pozici vůči pozici uživatele.
- `GameManager` - Provádí správu událostí, kontroluje spouštěče a volá akce. Zpracovává vstupy od uživatele, např. ukončení aplikace pomocí tlačítka zpět. Informuje řídící aplikaci o výsledku provedeného příkazu.

[1] Angl. Virtual Reality Modeling Language.
[2] Mezinárodní organizace pro normalizaci, angl. International Organization for Standardization.
[3] Angl. Extensible Markup Language.

- `LocationManager` - Získává GPS souřadnice zařízení, kterou dále využívají další manažeři.
- `PermissionManager` - Kontroluje, zdali dal uživatel souhlas s využitím komponent v zařízení důležitých pro chod aplikace (GPS pro získání polohy, kamery pro zobrazení scény).
- `UIManager` - Spravuje uživatelské rozhraní aplikace.



Obrázek 5.2: Schéma komunikace mezi manažery

5.3 Komunikace se serverem

Komunikace se serverem projektu EduARd probíhá přes tzv. REST API. REST neboli representational state transfer je architektura rozhraní poskytující standardy v komunikaci mezi systémy v distribuovaném prostředí. Hlavními charakteristikami této architektury jsou dělení odpovědností mezi klienty a servery a bezestavovost. Výhodou bezestavovosti (angl. statelesness) je schopnost porozumění jakékoliv zprávy bez znalosti předchozích zpráv [20]. Klient se na server dotazuje pomocí speciálních identifikátorů, tzv. URI[4], pomocí nichž získá přístup k určitým zdrojům serveru [15]. Pro komunikaci se využívá protokolu HTTP[5].

Přestože serverová část nabízí možnost ukládání i stahování dat, v této aplikaci bude implementována pouze jednostranná komunikace, konkrétněji pouze stahování. V této sekci budou uvedeny identifikátory využívané touto aplikací.

- `GET /api/Scenarios/{scenarioId}/download` - Stáhne soubor JSON popisující scénář s identifikátorem `scenarioId`. Tento identifikátor dostane aplikace v parametru příkazu od řídící aplikace (viz sekce 5.4).
- `GET /api/ScenarioAssets/{scenarioAssetId}` - Vrací seznam všech souborů potřebných pro načtení scénáře s identifikátorem `scenarioAssetId`[6].
- `GET /api/ScenarioAssets/{scenarioAssetId}/download` - Stáhne soubor s identifikátorem `scenarioAssetId`.

[4] Jednotný identifikátor zdroje, angl. Uniform Resource Identifier.
[5] Angl. Hypertext Transfer Protocol.
[6] Jméno identifikátoru je pravděpodobně chybné, měl by se jmenovat `scenarioId`.

5.4 Komunikace s řídící aplikací

Jak již bylo zmíněno v sekci 4.1, tato aplikace se bude řídit příkazy aplikace pro zobrazení výukových úloh (v tomto textu řídící aplikace). Pro správnou funkci této aplikace tedy bude nutné mít nainstalované obě zmíněné aplikace. Příkazy při spuštění zpracuje `DownloadManager` a v závislosti na jejich charakteru zavolá příslušné metody pro jejich vykonání. Při ukončení aplikace bude odeslána zpráva o výsledku vykonání příkazu zpět řídící aplikaci. Pokud tedy například dojde k chybě při vykonávání příkazu (např. při stahování dat), řídící aplikace bude moct na tento fakt zareagovat a pokusit se chybu odstranit (např. opětovným odesláním příkazu pro stažení dat, popř. upozorněním uživatele na připojení k internetu). V následujících podsekcích bude popsána implementace komunikace mezi těmito dvěma aplikacemi.

5.4.1 Zpracování příkazů a jejich typy

Komunikace s řídící aplikací je implementována pomocí tzv. `Intents`. `Intent` je třída obsahující abstraktní popis příkazu, který se má vykonat. `Intent` funguje jako propojení dvou aktivit (anglicky `Activity`), kde pod pojmem aktivita si lze zjednodušeně představit například aplikace spuštěné v zařízení. Řídící aplikace tedy v typickém případě spustí klientskou aplikaci (vytvoří novou aktivitu) a skrze `Intent` pošle příkaz, který má klientská aplikace vykonat, spolu se všemi parametry důležitými pro správné vykonání. V následujících bodech budou popsány všechny definované příkazy, jejich parametry a funkce.

- `download(token,scenarioId)` - Příkaz pro stažení dat scénáře s identifikátorem `scenarioId`. Parametr `token` představuje unikátní řetězec, pomocí něhož se klientská aplikace na serveru autentizuje a získá tak povolení k vykonání požadavku na serveru.
- `update(token,scenarioId)` - Příkaz pro aktualizaci dat scénáře s identifikátorem `scenarioId`.
- `erase_data(scenarioId)` - Příkaz pro smazání dat scénáře s identifikátorem `scenarioId`.
- `start_scenario(scenarioId)` - Příkaz pro načtení scénáře s identifikátorem `scenarioId`. Jelikož aplikace načítá jednotlivé scény ze scénáře, tento příkaz načte první scénu ve scénáři.
- `start_scene(scenarioId,sceneId)` - Příkaz pro načtení scénáře s identifikátorem `scenarioId` a scény s identifikátorem `sceneId`. Tento identifikátor značí pořadí scény ve scénáři.

Je nutné podotknout, že ačkoliv řídící i klientská aplikace jsou vyvíjeny pro platformu Android, obě aplikace jsou napsané v odlišných programovacích jazycích. Řídící aplikace využívá programovacího jazyka Java, klientská aplikace jazyka C#. Proto klientská využívá třídy `AndroidJavaObject` pro reprezentaci instance třídy `Intent`. Konverze mezi těmito třídami se pak provádí tímto způsobem:

```
AndroidJavaObject intent = currentActivity.Call<AndroidJavaObject>(
   "getIntent");
```

5.4.2 Odeslání zpětné vazby

Při ukončení klientské aplikace vždy dochází k odeslání zpětné vazby řídící aplikaci. Zpětná vazba obsahuje výsledek provedeného příkazu, z implementačního hlediska se jedná o jednoznačně určenou návratovou hodnotu. Tuto návratovou hodnotu řídící aplikace přijme, zpracuje a v závislosti na jejím charakteru provede další akce. Při bezchybném provedení příkazu klientská aplikace odešle hodnotu `100`, pokud dojde k chybě, odešle hodnotu `111`[7].

Pro komunikaci je opět využit `Intent`, způsob jeho odeslání se však liší od způsobu použitého při odesílání příkazu řídící aplikací. Zatímco řídící aplikace při odeslání příkazu zároveň spustí klientskou aplikaci, při odeslání zpětné vazby se řídící aplikace nespouští. Klientská aplikace se pouze ukončí a předá tak kontrolu řídící aplikaci. Je tedy nutné zajistit, aby se zpětná vazba odeslala ještě před ukončením klientské aplikace a zároveň zaručit ukončení klientské aplikace[8]. Z tohoto důvodu je pro odeslání zpětné vazby využit broadcast, který umožňuje implementaci výše popsané funkcionality se zajištěním daných kritérií. Řídící aplikace musí implementovat tzv. `BroadcastReceiver`, speciální komponentu umožňující zpracovat zprávu (návratovou hodnotu), kterou broadcast odešle. V klientské aplikaci je broadcast implementován jako tzv. plugin neboli zásuvný modul. Přestože je možné pracovat s třídami a objekty jazyka Java přímo v Unity (využívajícího jazyka C#), pro delší bloky kódu je tento přístup nevhodný a je účinější využít modulů, které jsou napsány v jazyce Java. Níže uvedený kód provádí odesílání zpětné vazby.

```
private Runnable sendData = new Runnable() {
    public void run() {
        // Create intent
        Intent sendIntent = new Intent();

        // Set flags, action and return value
        sendIntent.addFlags(
            Intent.FLAG_ACTIVITY_NO_ANIMATION|
            Intent.FLAG_FROM_BACKGROUND|
            Intent.FLAG_INCLUDE_STOPPED_PACKAGES
        );
        sendIntent.setAction("cz.cvut.fel.dcgi.eduard.
           sendintentplugin.toandroid");
        sendIntent.putExtra(Intent.EXTRA_TEXT, retVal);

        // Send broadcast
        sendBroadcast(sendIntent);
    }
};
```

[7] Výběr těchto návratových hodnot nemá žádné hlubší opodstatnění, důležité je, aby byly hodnoty odlišné a aby obě aplikace rozuměly jejich významu.

[8] Ukončení je nutné zajistit, protože klientská aplikace by případné další příkazy dokázala zpracovat pouze při novém spuštění.

5.5 Pozice scény

Výpočet pozice scény je proces skládající se ze tří kroků. Nejprve je nutné určit polohu zařízení v GPS souřadnicích. Dále je třeba vypočítat vzdálenost dvou bodů daných GPS souřadnicemi, kde prvním bodem je poloha zařízení a druhým bodem počátek scény[9], která bude vykreslena. Dle vypočítaných hodnot je poté nutné umístit tuto scénu do scény v Unity. Opakováním těchto kroků bude docházet k aktualizaci pozice scény vůči poloze uživatelova zařízení. Pro správnost vykreslování je nakonec potřeba správně rotovat scénu v závislosti na rotaci zařízení.

5.5.1 GPS lokalizace

GPS lokalizaci v aplikaci spravuje `LocationManager`. Ten využívá rozhraní `Input`, které nabízí Unity nativně. Toto rozhraní umožňuje mimo jiné i přístup k aktuálním GPS souřadnicím zařízení. Nutnou podmínkou je, aby zařízení mělo zapnutou GPS a aplikace měla povolení ji využívat. Práva k využití GPS zajišťuje `PermissionManager`. Aktualizace polohy zařízení je implementována jako tzv. koprogram (anglicky `Coroutine`). Koprogramy jsou speciální funkce schopné pozastavení vykonávání svého kódu. Nejsou tak vázané na aktualizace v každém snímku. Tento přístup umožňuje aktualizovat polohu zařízení pouze ve chvílích, kdy jsou tyto údaje dostupné. Pseudokód koprogramu zajišťujícího aktualizaci polohy zařízení je uveden níže.

```
Coroutine UpdateGPSPosition() {
    PermissionManager.CheckPermissions()
    EnableCompass()

    while (true)
        if (locationNotEnabled) yield break

        StartLocationService()

        // Wait until service initializes
        maxWaitingTime = 20
        while (LocationServiceInit && maxWaitingTime > 0)
            WaitForSeconds(1)
            maxWaitingTime--

        // Service didn't initialize in 20 seconds
        if (maxWaitingTime < 1) yield break

        // Connection has failed
        if (LocationServiceFailed)
            yield break

        // Success
        else
            UpdatePosition()
}
```

[9] Počátek scény je definován jeho kotvou, viz sekce 4.3.

5.5.2 Výpočet vzdálenosti dvou bodů daných GPS souřadnicemi

Vzdálenost dvou bodů daných GPS souřadnicemi lze vypočítat pomocí vzorce Haversine, který vypočítá délku ortodromy, tedy nejkratší spojnice dvou bodů na kulové ploše [4]. Tvar Země je tedy aproximován sférou[10]. Podle vzorce Haversine se délka ortodromy vypočítá následovně:

$$\begin{aligned}
&\Delta\phi = \phi_2 - \phi_1 \\
&\Delta\lambda = \lambda_2 - \lambda_1 \\
&a = sin^2(\Delta\phi/2) + cos\phi_1 \cdot cos\phi_2 \cdot sin^2(\Delta\lambda/2) \\
&c = 2 \cdot arctg2(\sqrt{a}, \sqrt{1-a}) \\
&d = R \cdot c,
\end{aligned}$$

kde, $\phi_{1,2}$ jsou zeměpisné šířky daných bodů, $\lambda_{1,2}$ zeměpisné délky daných bodů, $\Delta\phi$ rozdíl šířek bodů, $\Delta\lambda$ rozdíl délek bodů, R poloměr Země a d výsledná délka ortodromy. Je nutné zdůraznit, že vzdálenost bodů bude počítána zvlášť pro zeměpisnou šířku a délku. Z těchto údajů se pak dopočítají světové souřadnice ve scéně v Unity.

5.5.3 Umístění scény

Poté, co je vypočítána vzdálenost mezi dvěma body ve světě je nutné pomocí této informace umístit scénu do scény v Unity. Zde může být použití pojmu scéna matoucí. Scéna v Unity bude definována jako počítačová aproximace (model) reálného světa, kdežto scéna ve scénáři bude souborem objektů, které je třeba zobrazit v reálném světě. Tyto objekty budou umístěny do scény v Unity a obraz této scény se bude míchat s obrazem skutečného světa zachyceného kamerou zařízení. Cílem bude, aby umístění objektů v Unity odpovídalo umístění v reálném světě.

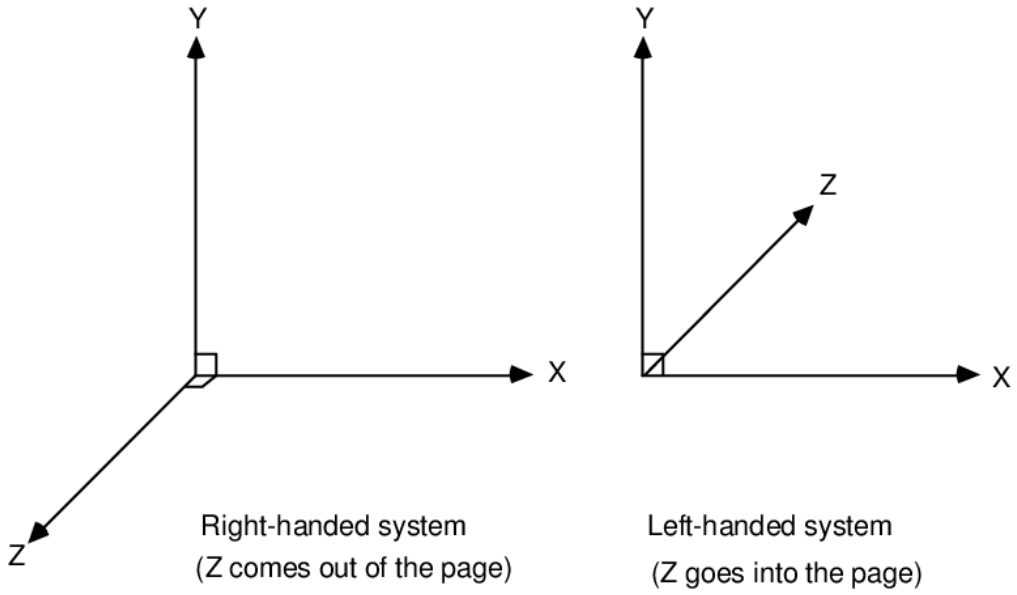
Kritériem použitelnosti bude, že tato aplikace bude vykreslovat pouze malé scény (z hlediska rozlohy, nikoliv škály). Důvodem tohoto kritéria je fakt, že scéna v Unity je ortogonální, kdežto reálný svět je zakřivený. Na malých vzdálenostech by se však chyba této aproximace neměla projevit. Proto bude reálný svět brán jako ortogonální, zeměpisná šířka bude představovat osu z a zeměpisná délka osu x v levotočivé souřadné soustavě, kterou Unity používá (pro demonstraci viz obrázek 5.3). Jednotlivé vzdálenosti šířek a délek mezi body pak budou souřadnicemi v daných osách. Z hlediska úplné korektnosti by též bylo vhodné využívat data o nadmořské výšce pro výpočet souřadnic osy y. Vzhledem k nepřesným datům naměřeným zařízením v terénu se však tento údaj využívat nebude, tj. počátek vykreslované scény bude mít souřadnici y vždy nulovou. Výpočet souřadnic počátku scény ze scénáře provádí `DataManager`, pseudokód funkce je uveden na následující straně.

[10]Budeme předpokládat, že v případech použití této aplikace, tj. na malých prostorách, se chyba aproximace při výpočtu pozice scény neprojeví.

```
function GetAnchorPosition(anchorLat, anchorLon) {
    // Calculate z coord from the distance between two latitudes
    if (anchorLat <= deviceLat)
        zPosition = -GetDistance(deviceLat, 0f, anchorLat, 0f)
    else
        zPosition = GetDistance(deviceLat, 0f, anchorLat, 0f)

    // Calculate x coord from the distance between two longitudes
    if (anchorLon >= deviceLon)
        xPosition = GetDistance(0f, deviceLon, 0f, anchorLon)
    else
        xPosition = -GetDistance(0f, deviceLon, 0f, anchorLon)

    return Vector3(xPosition, 0f, zPosition)
}
```



Obrázek 5.3: Porovnání pravotočivé a levotočivé souřadné soustavy[11].

5.5.4 Rotace scény

Pro správné zobrazení scény je nutné zajistit i její správnou rotaci. Pokud se tedy uživatel například otočí o 180 stupňů, scéna se musí otočit o stejný úhel. Pro detekci otáčení a měření úhlů naklonění se v zařízeních využívá gyroskopu. Gyroskop měří úhel naklonění okolo os x, y a z. Gyroskop je schopen měřit tyto úhly nezávisle na zrychlení a gravitaci (oproti např. akcelerometru) [30]. Přístup k údajům gyroskopu poskytuje v Unity rozhraní `Input`. V aplikaci se však při rotaci zařízení nerotuje se scénou, nýbrž s kamerou ve scéně. Tento přístup je implementačně jednodušší a provádí stejnou funkci. Rotace kamery se v kódu vypočítá následovně:

```
transform.localRotation = gyro.attitude * rot;
```

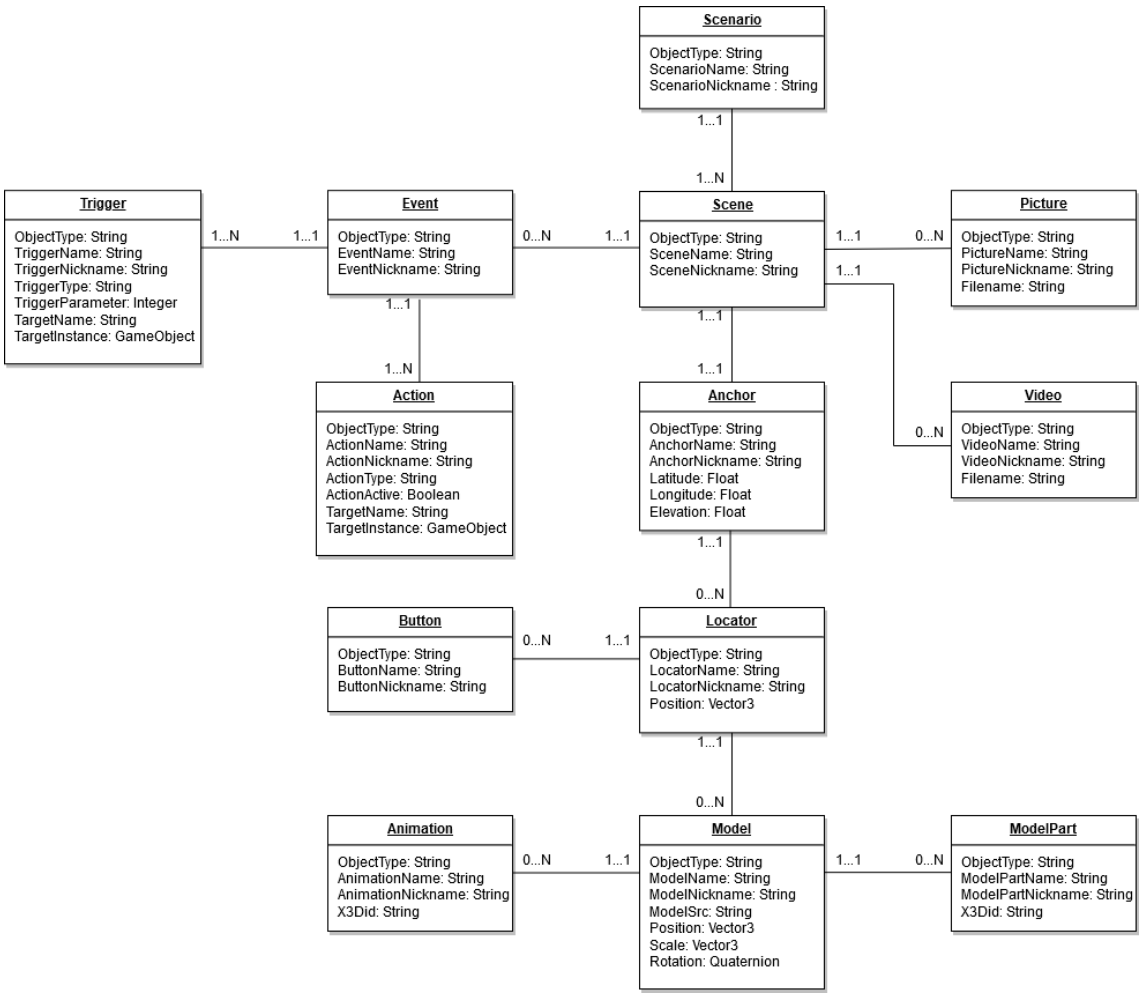
[11] Převzato z `https://www.researchgate.net/figure/Right-and-Left-Handed-Coordinate-systems_fig1_2457107`.

5.6 Načítání scénáře

Načítání scénáře v aplikaci spravuje `DataManager`. Načítání scénáře je dynamické, probíhá tedy za chodu aplikace. Z hlediska implementačního jej lze rozdělit do dvou částí: načtení souboru JSON obsahujícího popis scén ve scénáři a načtení souborů X3D obsahujících definice objektů (a animací). Následující podsekce se budou zabývat jednotlivými kroky.

5.6.1 Načítání souboru JSON

Načítání souboru JSON je iterativní, jelikož jeho hierarchie i typy objektů jsou předem známy (viz sekce 4.3). Pro načtení souboru JSON byl vytvořen skript `JSONParser`. Pro každý typ objektu byla vytvořena třída s odpovídající strukturou daného objektu. Každá třída tedy obsahuje proměnné, které uchovávají data daného objektu ze souboru JSON. Aby byla zachována hierarchie scénáře, uchovávají si třídy odkazy na své potomky, například kotva is uchovává seznam svých lokátorů, ty si uchovávají seznam svých modelů atd. `DataManager` tak pro načtení scény potřebuje pouze odkaz na kořenový uzel ve scénáři. Na obrázku 5.4 je zobrazen diagram tříd reprezentujících jednotlivé typy objektů souboru JSON.



Obrázek 5.4: Diagram tříd reprezentujících jednotlivé typy objektů souboru JSON

Načítání událostí

Součástí souboru JSON jsou i definice událostí. Události jsou reprezentovány třídou `Event`, uchovávají si seznam spouštěčů a akcí (reprezentované třídami `Trigger` a `Action`). Události nejsou načítány jako objekty ve scéně, jejich spouštěče a akce jsou však s některými objekty ve scéně spjaty. Jelikož jsou objekty jednoznačně identifikovatelné svým názvem (viz sekce 4.3), udržuje si `DataManager` mapu referencí na objekty ve scéně. Při načítání událostí si každá z instancí tříd `Trigger` a `Action` v mapě najde odkaz na příslušný objekt, ke kterému se váže. Při kontrole spouštěčů (např. vzdálenosti od objektu) a volání akcí (např. spuštění animace), jsou kontrola a akce provedeny na příslušných objektech. Seznam spouštěčů a akcí je uveden v sekci 4.5.

Načítání videí, obrázků a tlačítek

V souladu se způsobem načítání událostí je nutné každé video, obrázek a tlačítko ve scéně reprezentovat jako samostatný objekt. Z tohoto důvodu byl v Unity využit `Prefab`. `Prefab` umožňuje ukládat objekt se všemi jeho komponentami a nastaveními a snadno pak vytvářet instance tohoto objektu. Reference na jednotlivé instance těchto objektů jsou poté uloženy do mapy objektů, které spravuje `DataManager`. Podobou uživatelského rozhraní se dále bude zabývat sekce 5.7.

5.6.2 Načítání souboru X3D

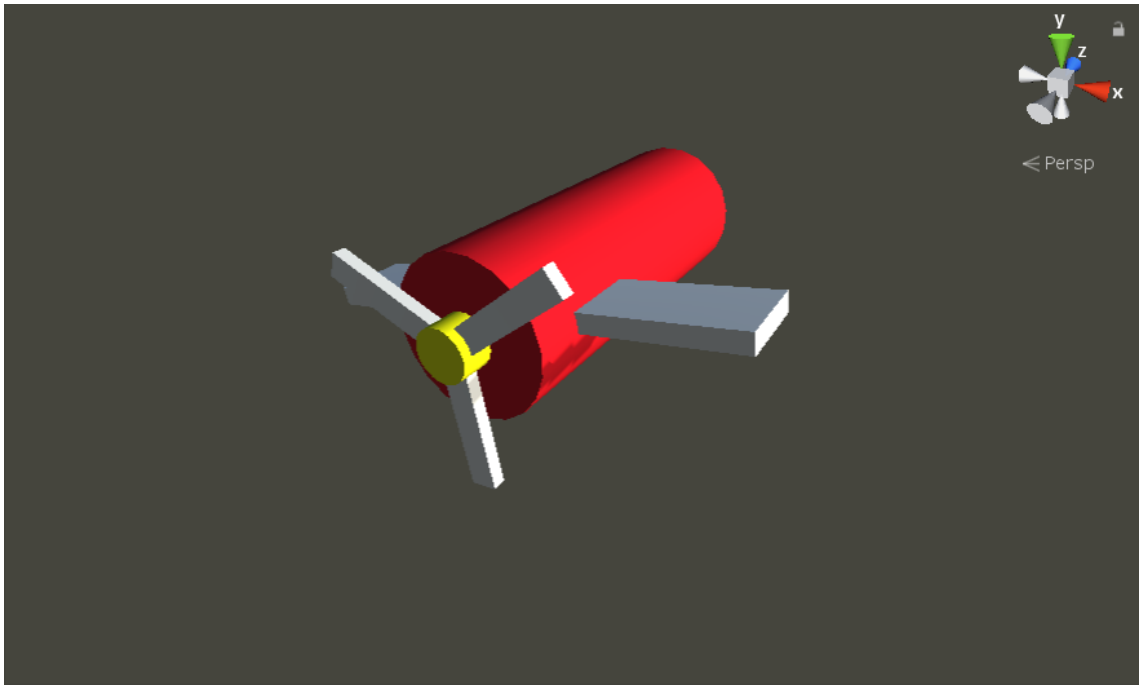
Jak bylo zmíněno v sekci 4.4, klientská aplikace načítá pouze modely a animace definované v souborech X3D. Proto bude některé uzly (např. uzly definující kameru či osvětlení) při načítání ignorovat. Pro načítání souborů X3D byl využit volně dostupný skript `UnityX3D`[12]. Tento skript byl pro účely tohoto projektu modifikován. Pro načítání jednotlivých uzlů je využito rekurze, jelikož hierarchie těchto uzlů není předem známa. Rekurzivní přístup dále zajišťuje, že načtený model v Unity si zachová hierarchii ze souboru X3D.

Načítání modelů

Aplikace je schopna načítat modely definované dvěma způsoby: pomocí primitiv a přes množinu indexovaných plošek (indexed face set). Z primitiv je možné načítat kvádry, sféry, a válce. Jedná se o průnik primitiv definovaných v X3D a Unity. Ukázka objektu tvořeného pouze z primitiv je na obrázku 5.5.

Modely definované přes množinu indexovaných plošek definují množinu vrcholů, normál a plošek, popřípadě i texturovacích souřadnic. Jelikož jsou polygonové sítě (polygon meshes) v Unity definované podobných způsobem, import modelů z X3D je přímočarý. Jediným rozdílem je definice plošek. Ty je v Unity vhodné definovat prostřednictvím trojúhelníků. V X3D je možné definovat plošky i jinými mnohoúhelníky, proto je nutné při načítání všechny plošky definované většími mnohoúhelníky rozdělit na trojúhelníky.

[12] Jedná se o skript pro import a export X3D scén do (z) Unity, viz `https://github.com/thefranke/unityx3d`. Je nutné podotknout, že nynější verze tohoto skriptu neodpovídá verzi, ze které se původně vycházelo.



Obrázek 5.5: Model tvořený z primitiv

Načítání materiálů

Osvětlovací model v X3D vychází z Phongova osvětlovacího modelu. Materiály jsou proto definované pomocí tří základních složek - difúzní, spekulární a ambientní (popř. i emisivní). Unity však využívá PBR[13] materiálů, které lépe fyzikálně simulují odraz světla. Při načítání materiálů je tedy nutné zajistit i co nejpřesnější přechod mezi těmito dvěma modely. V klientské aplikaci byla metoda konverze materiálů silně inspirována již existujícím řešením z původního skriptu `UnityX3D`. Tato metoda je popsána v tabulce 5.1. Ukázky modelu načteného v X3D prohlížeči a Unity jsou pro porovnání zobrazeny na obrázcích 5.6 a 5.7, příklad materiálu použitého v Unity pak na obrázku 5.8. Pro materiály v Unity byl zvolen standardní shader.

X3D	**Unity**
$\frac{\text{Difúzní}}{1-\|\|\text{Spekulární}\|\|}$	Difúzní
Spekulární	Spekulární
Ambientní	Nevyužito[14]
Emisivní	Emisivní
$\sqrt{\text{Lesk}}$ (Shininess)	Hladkost (Smoothness)

Tabulka 5.1: Tabulka konverze mezi materiály v X3D a Unity[15].

[13] Angl. physically based rendering.
[15] Unity nevyužívá ambientní složky v materiálech.
[15] Jednotlivé složky odpovídají složkám materiálů daných osvětlovacích modelů.

Obrázek 5.6: Model načtený v X3D prohlížeči

Obrázek 5.7: Model načtený v Unity

Obrázek 5.8: Materiál použitý v Unity (roh nosorožce)

Načítání animací

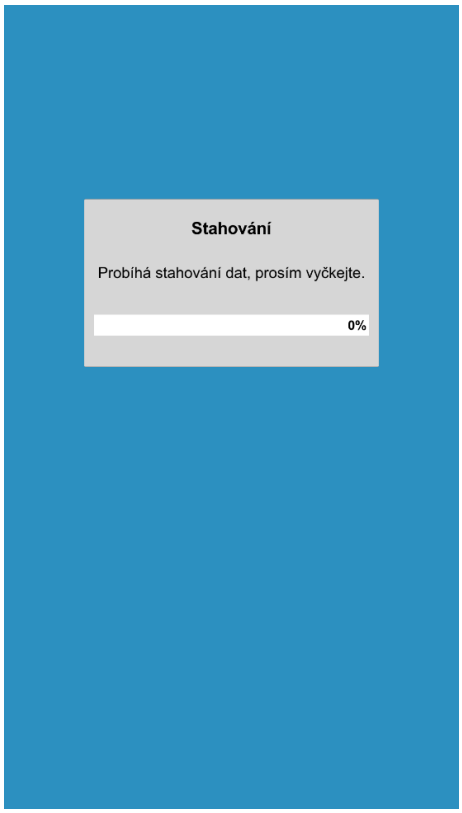
Ve formátu X3D jsou animace definovány pomocí interpolátorů. Interpolátory definují animační křivku pomocí seznamu klíčů a hodnot. Pro načítání interpolátorů byla vytvořena ekvivalentní třída `Interpolator`. Jedná se o abstraktní třídu, z níž dědí další třídy implementující konkrétní typy interpolátorů. Třída `Interpolator` obsahuje proměnou třídy `Animation`, do níž je uložena animační křivka z X3D. Skrze tuto třídu jsou animace přehrávány. Typy interpolátorů implementovaných v projektu odpovídají typům interpolací uvedených v sekci 4.5.

5.7 Uživatelské rozhraní

Následující sekce se bude zabývat implementací uživatelského rozhraní. Dle stanovených cílů (sekce 1.2) bylo uživatelské rozhraní navrženo a implementováno s důrazem na jednoduchost a srozumitelnost. Správu uživatelského rozhraní provádí `UIManager`.

Stahování dat

Při stahování dat se uživateli zobrazí panel stahování s indikátorem průběhu (obrázek 5.9).



Obrázek 5.9: Panel stahování s indikátorem průběhu

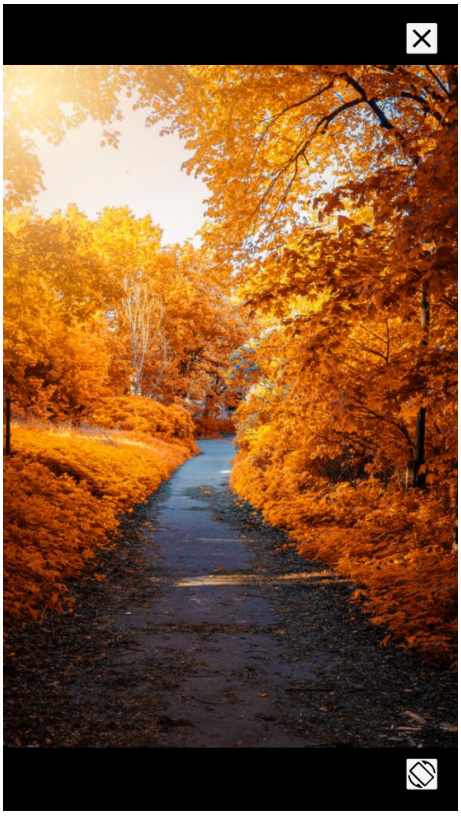
Vykreslování scény

Ve chvíli, kdy je scéna kompletně načtena a správně umístěna podle GPS souřadnic se začne vykreslovat v reálném světě. V této fázi se uživateli v rámci uživatelského rozhraní zobrazí pouze panel tlačítek (obrázek 5.10). Tlačítka slouží jako spouštěče, pomocí nichž lze například spouštět animace, přehrávat videa či zobrazit obrázky. Pro případ, že by se všechna tlačítka nevešla na displej, je panel tlačítek posuvný.

Přehrávání videí a zobrazování obrázků

Přehrávání videí a zobrazování obrázků jsou jedněmi z implementovaných akcí. Při těchto akcích se uživateli zobrazí samostatné panely (obrázek 5.11). Tyto panely obsahují dvě tlačítka: jedno pro přepínání mezi režimy portrait a landscape (na výšku a na šířku) a druhé pro uzavření panelu.

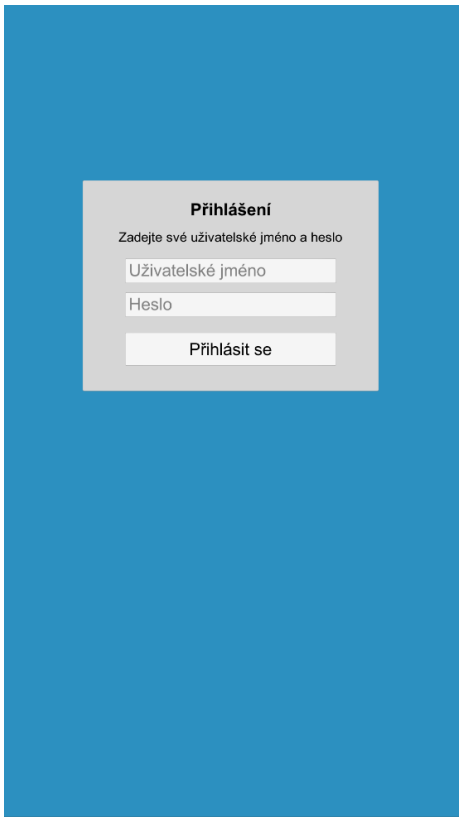
Obrázek 5.10: Panel tlačítek



Obrázek 5.11: Panel pro přehrávání videí a zobrazování obrázků

Přihlášení uživatele

Přihlášení uživatele v klientské aplikaci implementované není. Veškerá potřebná data jsou klientské aplikaci odeslána v příkazech řídící aplikací. Panel pro přihlášení (obrázek 5.12) se však uživateli zobrazí, pokud spustí aplikaci bez řídící aplikace (a tedy bez jakýchkoliv příkazů). Veškeré pokusy se přihlásit povedou na chybovou hlášku upozorňující uživatele, aby aplikaci spustil skrze řídící aplikaci.

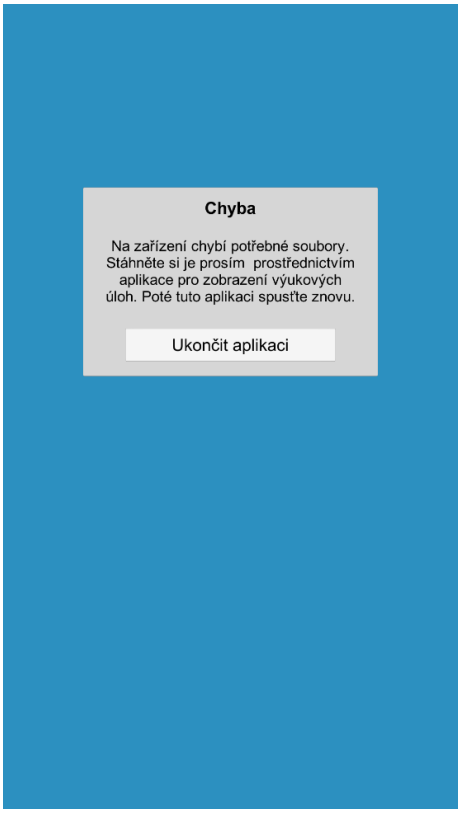


Obrázek 5.12: Panel pro přihlášení uživatele

Chybové hlášky

Chybová hláška se uživateli zobrazí, nastane-li chyba při vykonávání příkazu. Chybová hláška uživateli stručně popíše daný problém, tlačítkem "Ukončit aplikaci"aplikace uvědomí řídící aplikaci o chybě (viz sekce 5.4.2) a ukončí se. V následujících bodech bude uveden seznam všech typů chybových hlášek, příklad chybové hlášky je zobrazen na obrázku 5.13.

- Příkaz nebo parametr(y) příkazu jsou neplatné
- Připojení k internetu není k dispozici
- Server je nedostupný, nebo je dotaz na server neplatný
- Na zařízení chybí potřebné soubory
- Aplikace byla spuštěna bez příkazů



Obrázek 5.13: Příklad chybové hlášky

6 Testování

6.1 User-centered design

User-centered design (UCD, česky design zaměřený na uživatele) je metodika pro návrh, vývoj a testování softwaru. UCD vychází z oboru human-computer interaction (HCI, česky interakce člověk - počítač). Jejím cílem je definovat pravidla a postupy pro návrh a řízení vývoje softwaru za účelem splnění požadavků uživatelů z dané cílové skupiny. Základní myšlenkou UCD je umístit uživatele do centra vývoje produktu a reagovat na jeho námitky a požadavky a zamezit tak budoucím nesrovnalostem a konfliktům. Dodržováním pravidel této metodiky lze ušetřit čas při vývoji produktu a zaměřit se na definování důležitých aspektů a dodání funkčního řešení splňujícího očekávání uživatele [28].

Jelikož je tato aplikace jednou z komponent projektu EduARd, bylo důležité zajistit její správnou komunikaci s ostatními aplikacemi a komponentami (viz sekce 5.3 a 5.4). Během vývoje tedy byla aplikace průběžně testována ostatními členy vývojového týmu. Přestože je uživatelské rozhraní klientské aplikace velice jednoduché, na základě tohoto testování byly i tak přidány některé prvky uživatelského rozhraní, jmenovitě například chybové hlášky či indikátor průběhu při stahování dat (viz sekce 5.7). Testování s uživateli popsané v následujících sekcích je jedním z dalších nezbytných kroků této metodiky a její výsledky se promítnou do celkové podoby výsledného produktu.

6.2 Průběh testování

Testování proběhlo v Praze na Karlově náměstí v kampusu Fakulty elektrotechnické ČVUT (na dvoře před budovou E). Testování se zúčastnilo celkem pět participantů (účastníků). Aplikace byla testována na zařízení Xiaomi Redmi 5 Plus s operačním systémem Android 8.1. Testovací scéna se skládala ze sedmi modelů (nosorožce, natahovacího autíčka, letadla a čtyř šipek), jednoho videa a jednoho obrázku. Ke každému modelu byl přiřazen model šipky sloužící jako navigace při průchodu scénou (viz obrázek 6.1). Počátek scény byl indikován červenou šipkou. Cílem participantů bylo touto scénou projít postupně po zobrazených šipkách.

Údaje o participantech

Před testováním byl participantům předložen krátký dotazník. Cílem bylo získat základní informace o participantech a jejich zkušenostech s technologiemi rozšířené reality. V dotazníku byly položeny následující otázky:

1. Jaký je Váš věk?
2. Máte nějaké zkušenosti s rozšířenou realitou?
3. Znáte nějaké aplikace s rozšířenou realitou?
4. Používáte v současné chvíli nějaké aplikace s rozšířenou realitou?

Zaznamenané údaje o participantech na základě uvedeného dotazníku jsou uvedeny níže.

Participant 1

Věk: 23

Participant se setkal s rozšířenou realitou prostřednictvím demonstračních ukázek (např. na prezentacích společnosti Apple). Dříve hrál mobilní hru Pokémon GO, v současné době však žádnou aplikaci s rozšířenou realitou nepoužívá.

Participant 2

Věk: 24

Participant má zkušenosti především s mobilními hrami s rozšířenou realitou, např. Ingress Prime či Pokémon GO. Dále zmínil hry pro konzoli Playstation Vita. Momentálně žádné aplikace s rozšířenou realitou nepoužívá.

Participant 3

Věk: 22

Participant se s rozšířenou realitou setkal především na propagačních akcích. Žádné konkrétní aplikace však nezná a v současné době nepoužívá.

Participant 4

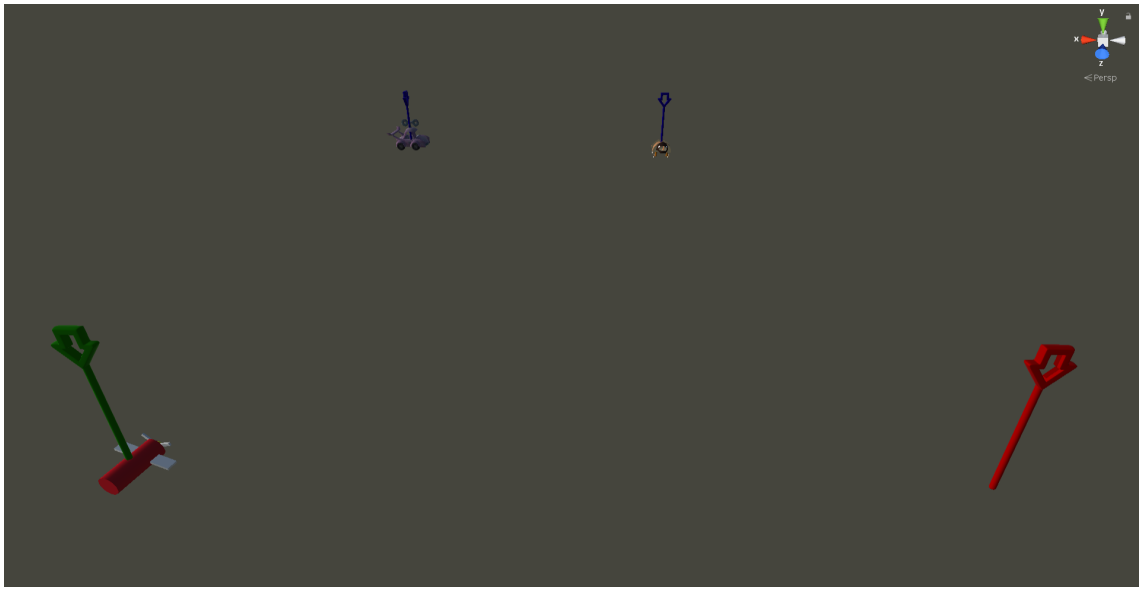
Věk: 23

Participant byl v minulosti hráčem mobilních her s rozšířenou realitou, jmenovitě například Pokémon GO či Harry Potter: Wizards Unite. V nynější době žádnou aplikaci s rozšířenou realitou nepoužívá.

Participant 5

Věk: 21

Participant se s rozšířenou realitou setkal prostřednictvím rozšířeně realitní aplikace k adventnímu kalendáři společnosti IKEA. Jiné aplikace nevyjmenoval, v současné době žádné jiné aplikace s rozšířenou realitou nepoužívá.



Obrázek 6.1: Náhled testovací scény v Unity

Úkoly

1. Stáhnout data ze serveru.
2. Spustit klientskou aplikaci a načíst scénu.
3. Dojít k počátku scény.
4. Projít scénu a u každého modelu spustit jeho animaci.
5. Přehrát video.
6. Zobrazit obrázek.
7. Ukončit aplikaci.

6.3 Výsledky

Participant 1

Participant neměl problémy se stažením dat ze serveru a následným spuštěním aplikace. Participant neměl problémy s pochopením významu šipek ve scéně a postupně je následoval. Problémem pro participanta bylo spouštění animací u jednotlivých modelů. Animace se pokoušel spustit klikáním na modely. U posledního modelu nebyla šipka zobrazena přesně nad ním.

Participant 2

Participant neměl problémy se stažením dat a spuštěním aplikace. Šipky ve scéně následoval postupně. Participant nevěděl, jak spouštět animace, hledal tlačítko pro spouštění animací v panelu tlačítek. U prvního modelu měl problémy s detekováním animace. Animace byla příliš jemná a z určitých úhlů špatně pozorovatelná. Kvůli nepřesnému určení polohy pomocí GPS docházelo i k nepřesnému vykreslování scény, vlivem čehož měl participant problémy s obcházením modelů. S přehráním videa a zobrazením obrázku participant problémy neměl, podotknul však, že ikony tlačítek pro přepínání mezi režimem na šířku a na výšku spíše připomínají ikony pro přepínání mezi celoobrazovkovým režimem a režimem v okně. Participant by též uvítal úvodní tutoriál vysvětlující základní ovládací prvky aplikace.

Participant 3

Participant neměl při průchodu scénou potíže s téměř žádným úkolem. Jediný problém mu činilo spouštění animací. Podobně jako u participanta 1 se snažil animace spustit klikáním na modely.

Participant 4

Participant neměl problémy se stažením dat a spuštěním aplikace, při průchodu scénou se navigoval pomocí šipek. Problém měl především se spouštěním animací, které se snažil spustit skrze panel tlačítek. U prvního modelu se mu špatně detekovala animace. Podobně jako u participanta 2 zaznamenal při testování nepřesnosti ve vykreslování scény způsobené

nepřesnou GPS lokalizací. S přehráním videa a zobrazením obrázku participant problémy neměl.

Participant 5

Participant neměl problémy se stažením dat a spuštěním aplikace, při průchodu scénou se navigoval pomocí šipek. Problémy mu činilo především spouštění animací. S přehráním videa a zobrazením obrázku problémy neměl.

6.4 Nálezy

V následující sekci bude uveden seznam všech nálezů. Nálezy se dělí podle závažnosti do tří kategorií: nízká, střední, vysoká. U každého nálezu bude uvedena jeho závažnost, popis problému a výčet participantů, kteří daný nález reportovali.

Nález 1

Závažnost: vysoká

Nalezeno u participantů: 1, 2, 3, 4, 5

Popis: Participanti nevěděli jak spouštět animace.

Nález 2

Závažnost: vysoká

Nalezeno u participantů: 2

Popis: Chybějící úvodní tutoriál.

Nález 3

Závažnost: střední

Nalezeno u participantů: 2, 4

Popis: Nepřesná GPS lokalizace. Výsledkem bylo nepřesné určování pozice scény vůči poloze uživatele.

Nález 4

Závažnost: střední

Nalezeno u participantů: 2, 4

Popis: Animace u prvního modelu (nosorožce) byla špatně detekovatelná.

Nález 5

Závažnost: nízká

Nalezeno u participantů: 1

Popis: U posledního modelu (letadla) se šipka ukazovala na špatném místě.

Nález 6

Závažnost: nízká

Nalezeno u participantů: 2

Popis: Ikony tlačítek pro přepínání mezi režimy na šířku a na výšku připomínají ikony pro přepínání mezi celoobrazovkovým režimem a režimem v okně.

6.5 Revize na základě nálezů

V následující sekci bude uvedeno řešení nálezů ze sekce 6.4. U každého nálezu bude uvedena jeho závažnost, popis a řešení.

Nález 1

Závažnost: vysoká

Popis: Participanti nevěděli jak spouštět animace.

Řešení: Do aplikace byl přidán tutoriál (viz obrázek 6.2) popisující základní ovládání a funkcionality aplikace, včetně spouštění animací.

Obrázek 6.2: Ukázka tutoriálu

Nález 2

Závažnost: vysoká

Popis: Chybějící úvodní tutoriál.

Řešení: Do aplikace byl přidán tutoriál (viz řešení nálezu 1).

Nález 3

Závažnost: střední

Popis: Nepřesná GPS lokalizace. Výsledkem bylo nepřesné určování pozice scény vůči poloze uživatele.

Řešení: Přesnost GPS závisí na mnoha faktorech (viz sekce 2.4). Tento problém bohužel nebyl zcela vyřešen a přetrvává tak i v poslední verzi.

Nález 4

Závažnost: střední

Popis: Animace u prvního modelu (nosorožce) byla špatně detekovatelná.

Řešení: Jednalo se o animaci otáčení hlavy. Animace byla zvýrazněna (hlava se otáčí o větší úhel).

Nález 5

Závažnost: nízká

Popis: U posledního modelu (letadla) se šipka ukazovala na špatném místě.

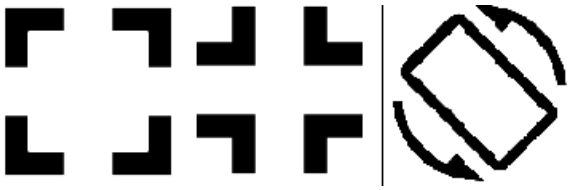
Řešení: Pozice šipky byla opravena.

Nález 6

Závažnost: nízká

Popis: Ikony tlačítek pro přepínání mezi režimy na šířku a na výšku připomínají ikony pro přepínání mezi celoobrazovkovým režimem a režimem v okně.

Řešení: Ikony byly nahrazeny novou ikonou lépe vystihující funkci tlačítek. Porovnání starých ikon a nové ikony je zobrazeno na obrázku 6.3.



Obrázek 6.3: Porovnání starých ikon (vlevo) a nové ikony (vpravo)

7 Závěr

Cílem této práce bylo vytvořit návrh a implementaci mobilní klientské 3D aplikace s rozšířenou realitou využívající GPS lokalizaci. V kapitole 3 byla nejprve popsána provedená rešerše dostupných nástrojů a projektů zabývajících se podobnou problematikou. Na základě nasbíraných dat bylo pro projekt zvoleno SDK Vuforia a herní engine Unity. Dále byl v kapitole 4 uveden návrh aplikace zabývající se její strukturou a návrhem jejích jednotlivých částí. Kapitola 5 poté vnesla náhled do metod použitých při implementaci projektu. Výsledná aplikace byla následně otestována s uživateli (kapitola 6) a na základě výsledků z testování upravena do současné podoby.

Výsledkem této práce je funkční prototyp klientské aplikace využívající rozšířené reality a GPS lokalizace. Aplikace je schopna vykreslovat scény vytvořené editorem 3D scén. Tyto scény jsou vykreslovány v závislosti na poloze uživatele dané GPS souřadnicemi. Veškerá data si aplikace stahuje ze serveru, příkazy přijímá od aplikace pro zobrazení výukových úloh. Aplikace je kompatibilní se zařízeními s operačním systémem Android 4.1 Jelly Bean a vyšší obsahujícími GPS lokátor a gyroskop.

V nynějším stavu však aplikace není zcela bezproblémová. Jejím hlavním nedostatkem je pravděpodobně samotná GPS lokalizace, jedna z jejích klíčových vlastností. Již během vývoje a poté především při testování se projevila nepřesnost určování polohy pomocí GPS. Tento jev může ztížit průchod scénou, především pak při prohlížení jednotlivých modelů. Pro budoucí vývoj aplikace bude vhodné diskutovat způsoby mitigace tohoto problému, například využitím technologie SLAM pro mapování prostředí. Dále, jak již bylo zmíněno v sekci 3.4, bude nutné do aplikace dodat detekci 2D značek. Vzhledem k vybranému SDK věřím, že je projekt dobře připraven pro tento typ rozšíření.

8 Literatura

[1] 12 Best Augmented Reality SDKs [online]. [cit. 10.10.2019]. Dostupné z: https://dzone.com/articles/12-best-augmented-reality-sdks.

[2] ARCore - Working with anchors [online]. [cit. 5.10.2019]. Dostupné z: https://developers.google.com/ar/develop/developer-guides/anchors.

[3] ARCore Location – Android Studio [online]. [cit. 5.10.2019]. Dostupné z: https://www.appoly.co.uk/2018/03/28/arcore-location/.

[4] Calculate distance, bearing and more between Latitude/Longitude points [online]. [cit. 20.11.2019]. Dostupné z: https://www.movable-type.co.uk/scripts/latlong.html.

[5] Get Ready for ARKit 3 [online]. [cit. 13.10.2019]. Dostupné z: https://developer.apple.com/augmented-reality/arkit/.

[6] How does GPS work? [online]. [cit. 28.9.2019]. Dostupné z: http://www.physics.org/article-questions.asp?id=55.

[7] How GPS Works [online]. [cit. 28.9.2019]. Dostupné z: https://www.gps.gov/multimedia/poster/poster.txt.

[8] Ingress: Everything you need to know in 2019 [online]. [cit. 10.10.2019]. Dostupné z: https://www.androidcentral.com/ingress.

[9] Introducing JSON [online]. [cit. 17.11.2019]. Dostupné z: http://www.json.org/.

[10] Mapbox - Documentation [online]. [cit. 5.10.2019]. Dostupné z: https://docs.mapbox.com/.

[11] Pokemon GO - Apps on Google [online]. [cit. 6.10.2019]. Dostupné z: https://play.google.com/store/apps/details?id=com.nianticlabs.pokemongo.

[12] SMART - Seamless AR Tracking [online]. [cit. 16.10.2019]. Dostupné z: https://www.wikitude.com/smart-augmented-reality/.

[13] The Ultimate Guide to Understanding Augmented Reality (AR) Technology [online]. [cit. 1.10.2019]. Dostupné z: https://www.realitytechnologies.com/augmented-reality/.

[14] The Ultimate Guide to Understanding Virtual Reality (VR) Technology [online]. [cit. 2.10.2019]. Dostupné z: https://www.realitytechnologies.com/virtual-reality/.

[15] Uniform Resource Identifiers (URIs) [online]. [cit. 17.11.2019]. Dostupné z: https://developer.yahoo.com/social/rest_api_guide/uri-general.html.

[16] Unity AR+GPS Location [online]. [cit. 5.10.2019]. Dostupné z: https://docs.unity-ar-gps-location.com/#main-features.

[17] Unity Manual - About AR Foundation [online]. [cit. 13.10.2019]. Dostupné z: https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@2.2/manual/index.html.

[18] Vortex Planetarium for Android - Features [online]. [cit. 6.10.2019]. Dostupné z: http://www.vortexplanetarium.com/features/.

[19] What is Augmented Reality (AR) and How does it work [online]. [cit. 1.10.2019]. Dostupné z: https://thinkmobiles.com/blog/what-is-augmented-reality/.

[20] What is REST? [online]. [cit. 17.11.2019]. Dostupné z: https://www.codecademy.com/articles/what-is-rest.

[21] What is X3D? [online]. [cit. 5.10.2019]. Dostupné z: http://www.web3d.org/x3d/what-x3d/.

[22] Wikitude Augmented Reality SDK [online]. [cit. 16.10.2019]. Dostupné z: https://www.wikitude.com/products/wikitude-sdk/.

[23] Steve Aukstakalnis. *Practical Augmented Reality: A Guide to the Technologies, Applications, and Human Factors for AR and VR*. Addison-Wesley, 2016.

[24] Joseph Hocking. *Unity in Action. Multiplatform game development in C#*. Manning Publications Co., 2018.

[25] Dieter Schmalstieg, Tobias Höllerer. *Augmented Reality: Principles and Practice*. Addison-Wesley, 2016.

[26] Paul Milgram, Haruo Takemura, Akira Utsumi, Fumio Kishino. Augmented reality: A class of displays on the reality-virtuality continuum. *Telemanipulator and Telepresence Technologies*, 1994. [online] [cit. 1.10.2019]. Dostupné z: http://etclab.mie.utoronto.ca/publication/1994/Milgram_Takemura_SPIE1994.pdf.

[27] Micheal Lanham. *Augmented Reality Game Development*. Packt Publishing Ltd., 2017.

[28] Travis Lowdermilk. *User-Centered Design: A Developer's Guide to Building User-Friendly Applications*. O'Reilly Media, 2013.

[29] Pavel Tišnovský. XML + 3D = X3D [online]. [cit. 5.10.2019]. Dostupné z: https://www.root.cz/clanky/xml-3d-x3d/.

[30] Jakub Čížek. Pojďme programovat elektroniku: Jak vlastně funguje akcelerometr a gyroskop nejen ve vašem telefonu [online]. [cit. 21.11.2019]. Dostupné z: https://www.zive.cz/clanky/pojdme-programovat-elektroniku-jak-vlastne-funguje-akcelerometr-a-gyroskop-nejen-ve-vasem-telefonu/jak-funguje-gyroskop/sc-3-a-194858-ch-114926/default.aspx#articleStart.

Příloha A

Seznam použitých zkratek

2D - dvoudimenzionální

3D - trojdimenzionální

API - application programming interface

AR - augmented reality

AV - augmented virtuality

ČVUT - České vysoké učení technické

GPS - Global Positioning System

HCI - human-computer interaction

HMD - head-mounted display

HTTP - Hypertext Transfer Protocol

ISO - International Organization for Standardization

JSON - JavaScript Object Notation

MR - mixed reality

OS - operační systém

PBR - physically based rendering

REST - representational state transfer

RPG - role-playing game

RV kontinuum - realitně-virtuální kontinuum

SDK - software development kit

SLAM - simultaneous localization and mapping

SMART - seamless augmented reality tracking

UCD - user-centered design

URI - Uniform Resource Identifier

VR - virtual reality

VRML - Virtual Reality Modeling Language

X3D - Extensible 3D

XML - Extensible Markup Language

Příloha B

Seznam X3D uzlů

Následující tabulka uvádí všechny uzly souboru X3D a jejich atributy (popř. hodnoty atributů), které lze v klientské aplikaci načítat. Je nutné podotknout, že skript pro načítání souborů X3D je volně dostupný[1] a pro účely tohoto projektu byl pouze modifikován (viz sekce 5.6.2). Načítání některých uzlů již tedy bylo předem implementováno a dodáno bylo pouze načítání chybějících uzlů, popř. bylo načítání některých uzlů modifikováno. Uzly, jejichž načítání nebylo v původním skriptu a nebo bylo modifikováno, jsou vypsány tučně.

Uzel	**Atributy (hodnoty atributů)**
Appearance	
Box	size
Color	color
ColorInterpolator	DEF, keyValue, key, toField (diffuseColor, specularColor, emissiveColor)
CommonSurfaceShader	DEF, USE, diffuseFactor, specularFactor, emissionColor, shininessFactor
Coordinate	DEF, USE, point
Cylinder	radius, height
Group	
ImageTexture, ImageTexture2D	url, containerField (diffuseTexture, normalTexture, specularTexture, emissiveTexture)
IndexedFaceSet	coordIndex, texCoordIndex
Material	DEF, USE, diffuseColor, specularColor, emissiveColor, shininess
MultiTextureCoordinate	
Normal	vector
OrientationInterpolator	DEF, keyValue, key
PositionInterpolator	DEF, keyValue, key, toField(set_translation, translation, set_scale, scale, size)
Route	fromNode, toNode, toField

[1] Jedná se o skript pro import a export X3D scén do (z) Unity, viz `https://github.com/thefranke/unityx3d`. Je nutné podotknout, že nynější verze tohoto skriptu neodpovídá verzi, ze které se původně vycházelo.

Uzel	**Atributy (hodnoty atributů)**
ScalarInterpolator	DEF, keyValue, key, toField(transparency, shininess, radius, height)
Shape	
Sphere	radius
SurfaceShaderTexture	containerField (diffuseTexture, normalTexture, specularTexture, emissiveTexture)
TextureCoordinate	point
TimeSensor	cycleInterval, loop, enabled
Transform	DEF, translation, rotation, scale, center

Tabulka B.1: Seznam X3D uzlů

Příloha C

Obsah přiloženého CD

Složka	**Obsah**
Source	Zdrojové kódy projektu
Source - SendIntent plugin	Zdrojové kódy zásuvného modulu pro broadcast
Build - Client application	Instalační soubor klientské aplikace
Build - Control application	Instalační soubor aplikace pro zobrazení výukových úloh
Testing scenario assets	Soubory testovacího scénáře
Screenshots	Ukázky z aplikace
Videos	Videoukázky z aplikace
Thesis	Text bakalářské práce
ReadMe	Manuál